Hochschule für Technik und Wirtschaft Dresden
Fakultät Maschinenbau

HTWD

Aufgabenblatt für die Diplomarbeit

im Studiengang / Studienrichtung **Fahrzeugtechnik / Kraftfahrzeugtechnik**

Name der Diplomandin / Matrikel-Nr.: **Lingzhi Wu / 51846**

Thema **Entwicklung eines Mehrziel-Verfolgungsalgorithmus' auf Basis von LiDAR**

Lidar-Sensoren ermöglichen eine sehr gute Umfelddetektion. Häufig liefern die verwendeten Algorithmen unterschiedliche Ergebnisse für verschiedene Sensoren. Im Rahmen der Diplomarbeit soll ein auf RFS (Radom Finite Sets) basierender Algorithmus auf seine Zuverlässigkeit bei der Verfolgung verschiedener Objekte und für unterschiedliche Sensoren hin untersucht werden. Es sollen damit neue und stabile Vorhersage- und Warnalgorithmen entwickelt werden.

Arbeitspunkte:

- Einarbeitung in die Lidar-Technologie für unterschiedliche Messverfahren
- Durchführung von Messungen und Aufbereitung der Messdaten für einen automatisierten Vergleich
- Entwicklung und Anpassung des Detektionsalgorithmus' zur Erkennung typischer Straßenverkehrsobjekte
- Bewertung der Robustheit der Detektionsergebnisse für unterschiedliche Situationen

Die Ergebnisse der Arbeit sind in einem Poster zu dokumentieren.

Betreuer HTW: Prof. Dr. rer. nat. Toralf Trautmann
Dipl.-Ing. (FH) Franziskus Mendt

Ausgehändigt am: 15.04.2024 Einzureichen bis: 16.09.2024

.....
Prof. Dr. rer. nat. Toralf Trautmann
Verantwortlicher Hochschullehrer

.....
Prof. Dr.-Ing. Thomas Himmer
Prüfungsausschussvorsitzender

Hinweise zum Erstellen der Diplomarbeit unter
https://www.htw-dresden.de/hochschule/fakultaeten/maschinenbau/studium/diplomsemester
Die Aufgabenstellung kann nach Absprache und dem Vorliegen von Teilergebnissen erweitert bzw. eingeengt werden.

Hochschule für Technik und Wirtschaft Dresden
University of Applied Sciences

Diplomarbeit

Entwicklung eines Mehrziel-Verfolgungsalgorithmus‘ auf Basis von LiDAR

vorgelegt von:	Lingzhi Wu
Ort:	HTW Dresden
Studiengang:	Kraftfahrzeugtechnik
Betreuer:	Prof. Dr. rer. Nat. Toralf Trautmann
	Dipl.-Ing. (FH) Franziskus Mendt
Abgabedatum:	30.09.2024

Inhaltsverzeichnis

Abkürzungsverzeichnis

MTT	Multi-Target-Tracking
LiDAR	Light Detection and Ranging
RFS	Random Finit Sets
FISST	Finite Set Statistics
PGFLs	Probability Generating Functional
PDF	Probability Density Function
PHD	Probability Hypothesis Density
EM	Expectation-Maximization
GNN	Global Nearest Neighbor
MHT	Multiple Hypothesis Tracker
PMHT	probabilistische Multiple Hypothesis Tracking
JPDA	Joint Probabilistic Data Association
SIR	Sequential Importance Resampling
DST	Dempster-Shafer-Theorie
BPA	Basic Probability Assignment
Bel	Belief Function
Pl	Plausibility Function
MATLAB	Matrix Laoratory
FOV	Field of View
ROI	Reigion of Interest
GOSPA	Generalized Optimal Sub-Pattern Assignment
SC	switching component
SP	switching penalty
CT	Clustering Threshold

AT	Assignment Threshold
MCC	Min Num Cells per Cluster
CPU	Central Processing Unit
GPU	Graphics Processing Unit
FPS	Frames per Second
GPS	Global Positioning System

Verzeichnis der Formelzeichen und Symbole

Symbol	Einheit	Beschreibung
X	-	Zufällige endliche Menge von Objekten / eine Menge der Tracks
Z	-	Zufällige endliche Menge von Beobachtungen
Y	-	eine Menge der Wahrheiten
$\mathbb{R}$	-	Raum der reellen Zahlen
$\mathbb{X}$	-	Zustandsraum
$\mathbb{Z}$	-	Beobachtungsraum
k	-	Zeitschritt
$+$	-	Vorhergesagter Wert einer Variablen, Abkürzung für $k+1\|k$
x	m	Zustandsvektor
z	-	Beobachtungsvektor
v	km/h	Geschwindigkeitsvektor
ξ	-	Prozessrauschvektor
ζ	-	Beobachtungsrauschvektor
$f_{+/k}$	-	Zustandsübergangsgleichung
h	-	Beobachtungsfunktion
p	-	Wahrscheinlichkeitsdichtefunktion
q	-	Vorschlagsdichtefunktion
$p_o(o)$		Wahrscheinlichkeit des Belegungszustands *o*
$g(z\|x)$	-	Likelihood-Funktion der Beobachtung z gegeben den Zustand x
ν	-	Anzahl der Partikel
i	-	Index der Partikel
c		Index der Gitterzelle
w	-	Gewicht eines Partikels

$\widetilde{w}$	-	Unnormalisiertes Gewicht eines Partikels
p_S	-	fortdauernden Wahrscheinlichkeit
p_A	-	Assoziationswahrscheinlichkeit
D		vorhergesagte Wahrscheinlichkeitsdichtefunktion
δ	-	Dirac-Delta-Funktion
Ω	-	Universum-Menge
O	-	Belegung einer Gitterzelle
F	-	Freiheit einer Gitterzelle
A	-	beliebige Teilmenge des Rahmens
m	-	Basiswahrscheinlichkeitszuweisungsfunktion
Bel	-	Vertrauensfunktion
Pl	-	Plausibilitätsfunktion
r	-	Ordnung der Metrik
c	m	Grenzwert der Distanzschwelle
d_c	m	cutoff-basierte Distanz
d_b	m	Basisdistanz zwischen dem Track und der Wahrheit
α	-	Alpha-Parameter
h	-	Anzahl der nicht trivialen Zuordnungen
n_s	-	Anzahl der Umschaltungen
n_{miss}	-	Anzahl der verpassten Ziele
n_{false}	-	Anzahl der falschen Tracks

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

In den vergangenen Jahren hat sich das autonome Fahren dank der fortschreitenden technologischen Innovationen signifikant weiterentwickelt. Die Entscheidungen zu treffen und die Fahrsicherheit zu gewährleisten setzen dabei eine Verfügung über möglichst genaue Umgebungsinformationen voraus.

Die Multi-Target-Tracking-Technologie (MTT) steht beim Umfeldwahrnehmungssystem im Vordergrund. Auf funktionaler Ebene ermöglicht das MTT, mehrere dynamische Ziele aus Sensordaten zu identifizieren und danach kontinuierlich zu verfolgen. Zudem trägt sie zur Verbesserung der Umfeldwahrnehmung, der Routenplanung sowie der Entscheidungsfindung im autonomen Fahren bei. Als eine wichtige Handwaregrundlage für die Datenerfassung weist das LiDAR (Light Detection and Ranging) aufgrund seiner hochpräzisen und hochauflösenden Umfelderkennung eine einzigartige Stärke auf. Im Vergleich zu herkömmlichen Kameras kann es nicht nur unter schwierigen Lichtverhältnissen stabil arbeiten, sondern liefert auch präzise Daten zu Entfernungen und Positionen.

Allerdings ist die Mehrzielverfolgung weiterhin viele Herausforderungen konfrontiert, beispielsweise die Mehrzielerkennung und -vorhersage sowie die Zielverdeckung. Aus diesem Grund ist die Entwicklung eines robusten, umgebungsadaptiven und effizienten MTT-Algorithmus von großer Bedeutung.

Diese Diplomarbeit zielt sich auf die Entwicklung und Validierung vom effizienten Multi-Target-Tracking-Algorithmus auf Basis von LiDAR ab. Der in dieser Diplomarbeit verwendete Algorithmus basiert auf eine dynamische Gitterkarte in Kombination mit einem Partikelfilter, welches auf der RFS-Theorie (Random Finite Set) aufbaut und zusätzlich die Dempster-Shafer-Theorie (DST) integriert, um eine Mehrzielverfolgung und -vorhersage zu realisieren. Darüber hinaus ermöglicht er eine Analyse der Zieltrajektorien während der Verfolgung, selbst wenn ein Ziel vorübergehend verschwindet.

Die spezifischen Ziele lassen sich in die folgenden drei Teile unterteilen:

1. Datenverarbeitung

- Einarbeitung in die LiDAR-Technologie zur Anpassung an unterschiedliche Messverfahren.
- Durchführung von Messungen und Aufbereitung der Messdaten für einen automatisierten Vergleich.
- Entwicklung einer effektiven Methode zur Vorverarbeitung von LiDAR-Daten.

2. Multi-Target- Verfolgung

- Entwicklung eines robusten Algorithmus zur Multi-Target- Verfolgung, der in der Lage ist, in komplexen dynamischen Szenarien präzise mehrere Ziele zu verfolgen.

3. Validierung des Algorithmus und Leistungsbewertung

- Bewertung der Wirksamkeit und Robustheit des vorgeschlagenen Algorithmus durch Simulationen und Tests in realen Szenarien, um seine Anwendbarkeit in der Praxis sicherzustellen.

2 Grundlagen

Mit diesem Kapitel sollen die Grundlagen für eine mathematische Beschreibung des Algorithmus gelegt werden. Nachdem das MTT und deine Klassifikation erläutert wurden, werden die Statistik endlicher Mengen (Finite Set Statistics, FISST), die Partikelfilter-Theorie und die Dempster-Shafer-Theorie (DST) eingeführt.

Die Statistik endlicher Mengen (Finite Set Statistics, FISST) schafft einen mathematischen Rahmen für die Zustandsschätzung mehrerer dynamischer Objekte. In diesem Rahmen hat jeder Parameter seine eigene physikalische Bedeutung. Die Modellierung des Schätzproblems dynamischer Gitterkarten im Bereich der zufälligen endlichen Mengen ist vorteilhaft, weil sie die Bewältigung verschiedener komplexer Situationen fördert, in denen sich die Anzahl der Ziele ändert, wie zum Beispiel bei Neuentstehung, Verschwinden und Zusammenführung.

Durch die Kombination von Partikelfilter und DST lassen sich typische Verkehrsteilnehmer erkennen und verfolgen, indem die Belegungswahrscheinlichkeit und der Zustand jeder Zelle geschätzt werden.

2.1 Mehrzielverfolgung

MTT zielt darauf ab, den Zustand mehrerer Ziele wie Position, Geschwindigkeit usw. gleichzeitig zu erkennen und zu verfolgen. Dabei lassen sich die Probleme wie das Erscheinen und Verschwinden von Zielen und die Datenassoziation lösen.

Mehrzielverfolgung ist der Prozess, bei dem mehrere sich bewegende Ziele gleichzeitig verfolgt werden. Dies geschieht durch das Erkennen und Verknüpfen von Zielen in aufeinander folgenden Frames, um die Konsistenz der Identität jedes Ziels zu wahren. Die Aufgaben umfassen Zielerkennung, Datenassoziation, Vorhersage der Trajektorien, Identitätserhaltung und die Bewältigung von Verdeckungen. Mehrzielverfolgung findet breite Anwendung in Bereichen wie autonomem Fahren, Videoüberwachung und Roboternavigation, wobei die Algorithmen in der Lage sein müssen, in komplexen Umgebungen in Echtzeit und robust mit dynamischen Zielen umzugehen.

Der Prozess der Mehrzielverfolgung besteht typischerweise aus vier Hauptschritten: Zuerst werden in jedem Frame die Ziele durch Erkennungsalgorithmen identifiziert. Anschließend erfolgt die Datenassoziation, um die erkannten Ziele mit den Trajektorien des vorherigen Frames abzugleichen und ihre Identität beizubehalten. Danach wird die zukünftige Position der Ziele durch Vorhersagealgorithmen geschätzt. Abschließend umfasst das Management der Trajektorien die Initialisierung, Aktualisierung und Beendigung von Zielen.

2.1.1 Grundlegende Klassifikation

Zielverfolgungsalgorithmen können je nach Arbeitsprinzip in generative und diskriminative Modelle unterteilt werden.

1. Generative Modelle

Die Algorithmen wie Partikelfilter, PHD-Filter, Multi-Bernoulli-Filter, den Expectation-Maximization (EM) Algorithmus in Punktwolken und zufällige endliche Mengen-Partikelfilter benutzen generative Modelle. Diese Methoden beginnen mit dem Aufbau eines Zielmodells oder der Extraktion von Zielmerkmalen. Anschließend führen sie eine Ähnlichkeitssuche nach diesen Merkmalen in den folgenden Frames durch, um die Zielverfolgung schrittweise zu realisieren. Mittels mathematisches Modelles beschreiben sie die zu verfolgende Ziele.

Die generative Modell-Zielverfolgungsalgorithmen versuchen, das Verhalten und die Position des Ziels durch das Lernen des Generierungsprozesses der Punktwolken zu modellieren. Diese Methoden erfordern in der Regel den Aufbau eines Generierungsmodells, um die Zustandsübergänge und das Beobachtungsmodell des Ziels zu beschreiben. Generative Modelle sind besonders gut geeignet für Aufgaben mit hoher Unsicherheit, wie z. B. Mehrzielverfolgung oder Szenarien mit schwerwiegenden Verdeckungen.

2. Diskriminative Modelle

Diskriminative Modelle berücksichtigen sowohl das Zielmodell als auch Hintergrundinformationen. Durch den Vergleich der Unterschiede zwischen Zielmodell und Hintergrundinformationen extrahieren sie das Zielmodell. [2]

2.1.2 Weitergehende Klassifikation

1. Methoden basierend auf eine Assoziation von Beobachtungen und Trajektorien

Diese Methoden zerlegen das Mehrzielverfolgungsproblem in mehrere Einzelzielverfolgung-Probleme. Dann ist ein konventionelle Einzelzielverfolger für jedes einzelne Ziel verwendbar.

Ihre Schlüssel liegt dabei in der Assoziation von Beobachtungen und Trajektorien durch explizite Zuordnungstechniken. Klassische Datenassoziationsmethoden umfassen Global Nearest Neighbor (GNN), Multiple Hypothesis Tracker (MHT) sowie probabilistische Multiple Hypothesis Tracking (PMHT) und Joint Probabilistic Data Association (JPDA). Diese Methoden sind jedoch hauptsächlich auf ideale Szenarien mit bekannter und fester Zielanzahl ausgelegt und haben Schwierigkeiten, sich an zufällige und unbekannte Zielzahlen anzupassen. In komplexen Szenarien wie niedrigen Signal-

Rausch-Verhältnissen und dichten Zielansammlungen sind die Verfolgungsleistungen dieser Methoden stark eingeschränkt.

2. Methoden ohne Assoziation von Beobachtungen und Trajektorien

Im Gegensatz zu den datenassoziationsbasierten Methoden, die das Problem durch Zerlegung angehen, konzentrieren sich diese Ansätze auf die Erweiterung der Filtertheorie selbst, um das Problem der Datenassoziation zu vermeiden. Der Kernansatz besteht darin, den Mehrzielzustand als ein einheitliches Ganzes zu schätzen.

Durch statistische und probabilistische Methoden werden Zielverfolgung und Datenfusion behandelt, damit sie von der schrittweisen Assoziation von Beobachtungen und Trajektorien erforderlich nicht abhängen. Die Methode von RFS zur Mehrzielfilterung verfolgt diesen Ansatz.

Das von Mahler vorgeschlagene FISST stellt eine tiefgreifende Erweiterung der Informations-fusionstechniken und des Filtertheorie-Rahmens dar. FISST beschreibt Zustands- und Beobachtungs-variablen durch RFS. Dabei werden Punktprozesse verwendet, um die Entstehung, Ableitung und Sterblichkeit von Zielen sowie den Beobachtungsprozess zu modellieren. Es werden separate Wahrscheinlichkeitsmodelle für Fehlalarme, Zielentwicklung und Beobachtungsprozesse aufgebaut.

Das probabilistische Generierungsfunktionsmodell (Probability Generating Functional, PGFLs) bietet eine kompakte Methode zur Problemlösung bei der Datenfusion. Auf Basis verschiedener Ziel- und Störmodelle können unterschiedliche Formen der Mehrzielübertragungsdichte und der approximativen bayesschen Posterior-Dichte abgeleitet werden. Dies ermöglicht die Entwicklung verschiedener BM-Filter-Iterationsgleichungen und vermeidet die explizite Datenassoziation von Beobachtungen und Trajektorien.

Als eine Technik zur approximativen bayesschen Filterung wurde der Partikelfilter auf den Bayer-Markov-Rahmen ausgeweitet, was zur Entwicklung des RFS-Filters führte.[2]

2.1.3 Schwerpunkte

Die Mehrzielverfolgung stellt eine der herausforderndsten Aufgaben im Bereich der komplexen Zustandsschätzung dar. Sie erfordert zum einen zuverlässige Systemmodellierung und Parameterabschätzung, zum anderen effektive Methoden zur Informationsfusion und Entscheidungsfindung. Angesichts komplexer Szenarien und steigender praktischer Anforderungen ist die Erforschung und Entwicklung eines robusten und effizienten MTT-Algorithmus eine vielversprechende und anspruchsvolle Aufgabe.

Darüber hinaus sollte auch die Integration der Datenanalyse auf niedriger Ebene, der Informationsverarbeitung und der strategischen Entscheidungsfindung erfolgen, wie beispielsweise die Klassifizierung komplexer Ziele, die Situationsbewertung, die Erkennung von Bedrohungsstufen und das Aktionsfeedback. Dies stellt nicht nur eine enorme Herausforderung dar, sondern eröffnet auch größere Anwendungsmöglichkeiten.

Die Schwierigkeiten der Objektverfolgung sind konkret in zwei Aspekte unterteilen. Aus der Sicht des beobachteten Objekts ist die Anzahl der Ziele im Laufe der Beobachtungszeit unbekannt. Das heißt, die Ziele können jederzeit auftreten oder verschwinden. Aus der Sicht des Beobachtungsprozesses erzeugt zunächst nicht jedes Ziel zu jedem Zeitpunkt eine entsprechende Beobachtung. Dies bietet eine Möglichkeit des Ausfalls der Beobachtungen. Darüber hinaus werden wegen der Störsignale nicht alle Beobachtungen von Zielen erzeugt. Weiterhin ist es schwierig, jede Beobachtung eindeutig einem Ziel zuzuordnen. Dazu wird die Datenassoziation normalerweise benutzt. [3]

2.2 FISST - Finite Set Statistics

In diesem Abschnitt wird das zuvor erwähnte RFS und FISST vorgestellt. Eine zufällige endliche Menge $\mathbf{X}$ ist eine auf dem Zustandsraum X definierte Zufallsmenge, wobei X der Raum $\mathbb{R}^n$ sein kann und den Zustand des Ziels wie Position, Geschwindigkeit usw. darstellt. Für einen gegebenen Zustandsraum X kann die endliche Zufallsmenge $\mathbf{X}$ als folgende Formal bezeichnet sein.

$$\mathbf{X} = \{x_1, x_2, \dots, x_n\} \tag{2.1}$$

Der Zustandsvektor jedes Ziels wird durch $x = [x, y, z, \dot{x}, \dot{y}, \dot{z}]$ definiert, der die Position und Geschwindigkeit des Ziels darstellt.

Die Wahrscheinlichkeitsdichtefunktion (Probability Density Function, PDF) einer endlichen Zufallsmenge werden verwendet, um die statistischen Eigenschaften des Zielsatzes zu beschreiben.

$$p(\mathbf{X}) = p(|\mathbf{X}|) \cdot p(\mathbf{X}||\mathbf{X}|) \tag{2.1}$$

$p(\mathbf{X})$ ist die Wahrscheinlichkeitsdichtefunktion (PDF) für die endliche Zufallsmenge $\mathbf{X}$.

$p(|\mathbf{X}|)$ ist die Wahrscheinlichkeitsdichtefunktion für den Betrag von $\mathbf{X}$.

$p(\mathbf{X}||\mathbf{X}|)$ ist die bedingte Wahrscheinlichkeitsdichtefunktion von $\mathbf{X}$ gegeben dem Betrag von $\mathbf{X}$.[4]

2.3 Partikelfilter-Theorie

Durch die Beobachtung einer Zufallsvariable erfolgt die Schätzung des Zustands dieser Zufallsvariable oder einer damit verbundenen Zufallsvariable, also die Filterung, was eines der Kernprobleme in der Signalverarbeitung darstellt. Solche Probleme treten auch in der Zielverfolgung auf. Die bayessche Schätzung ist als ein wichtiges theoretisches Werkzeug zur Lösung solcher Probleme betrachtet, die die theoretische Grundlage für das Partikelfilter bildet. Dabei wird eine Gruppe gewichteter Partikel verwendet, um die Wahrscheinlichkeitsverteilung der Zustands-variablen zu approximieren. Im Fall der iterativen Aktualisierung der Partikelgruppe ermöglicht eine rekursive bayessche Schätzung.

Der Partikelfilter ist eine wichtige nichtlineare rekursive bayessche Filtermethode, die relativ geringe Anforderungen an das System stellt. Es ist weder erforderlich, dass die Systemmodellgleichungen linear sind, noch dass das Systemrauschen normalverteilt ist. Deshalb hat er gute Erweiterbarkeit und Allgemeingültigkeit des Algorithmus zur Folge. Mit der Anwendung von FISST auf Mehrzielverfolgungsprobleme hat das Partikelfilter eine neue Entwicklungsstufe erreicht, was zur Entstehung des RFS-Filters geführt hat, der in dieser Diplomarbeit verwendet wird. [2]

2.3.1 Bayessche Schätzung

Kernprobleme wie Lokalisierung, Verfolgung und dynamische Parameterschätzung betreffen die rekursive Zustandsabschätzung, bei der durch die Sequenzbeobachtungen z_1, z_2, …, z_k zum diskreten Zeitpunkt k der Zustand x_1, x_2, …, x_k rekursiv geschätzt wird. Hierbei ist die Beobachtung z_k eine Funktion des Zustands x_k, und der Zustand x_k entwickelt sich im Laufe der Zeit typischerweise als diskreter oder kontinuierlicher Markov-Prozess.

$$x_k = f_k(x_{k-1}, \xi_k) \tag{2.3}$$

In der Gleichung steht k für die diskrete Zeit (Frame), $x_k \in \mathbb{R}^{n_x}$ repräsentiert den Zustand zur diskreten Zeit, $\xi_k \in \mathbb{R}^{n_\xi}$ ist das Prozessrauschen und f_k ist die diskrete Zustands-übergangsgleichung. Sensoren arbeiten in der Regel auf Basis einer diskreten Zeitabtastung. Daher wird die Systembeobachtung normalerweise durch eine diskrete Gleichung modelliert:

$$z_k = h_k(x_k, \zeta_k) \tag{2.4}$$

In der Gleichung steht $z_k \in \mathbb{R}^{n_z}$ für die Beobachtung zum Zeitpunkt k. $\zeta_k \in \mathbb{R}^{n_\zeta}$ ist das Beobachtungsrauschen. $h_k : \mathbb{R}^{n_z} \times \mathbb{R}^{n_\zeta} \to \mathbb{R}^{n_z}$ ist die Beobachtungsfunktion.

Zusammen beschreiben die Zustands- und Beobachtungsfunktionen das diskrete Zustandsraum-modell (State Space Model, SSM) für das rekursive Zustandsabschätzungsproblem. Der Filter liefert in der Regel die Filterverteilung als Ausgabe.[2]

Der Zustandsvektor $x_k \in \mathbb{X}$ beschreibt den n_x-dimensionalen Zustand eines Objekts zum Zeitpunkt k im Zustandsraum $\mathbb{X} = \mathbb{R}^{n_x}$. Der Bayes-Filter berücksichtigt n_x-dimensionale Beobachtungen $z_k \in \mathbb{Z}$ im Beobachtungsraum $\mathbb{Z} = \mathbb{R}^{n_x}$. Alle Beobachtungen bis zum Zeitpunkt k werden durch $z_{1:k} = \{z_1, \dots, z_k\}$ dargestellt. Die posteriori-Dichte ist die Wahrscheinlichkeits-dichtefunktion des Zustands x_k zum Zeitpunkt k, gegeben alle Beobachtungen $z_{1:k}$.[1]

$$p_k(x_k | z_{1:k}) \tag{2.5}$$

2.3.2 Partikelfilter

Der Partikelfilter stellt die posteriori Wahrscheinlichkeitsdichtefunktion $p(x_k)$, durch eine Menge von ν_k Partikeln und deren Gewichten $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{\nu_k}$ dar,

$$p(x_k|z_{1:k}) \approx \sum_{i=1}^{\nu_k} w_k^{(i)} \delta\left(x_k - x_k^{(i)}\right) \tag{2.6}$$

wobei $x_k^{(i)}$, $w_k^{(i)}$, ν_k jeweils den Partikelzustand zum Zeitpunkt k, das Gewicht und die Gesamtanzahl der Partikel darstellen.

δ ist die Dirac-Delta-Funktion, die die folgende Bedingung erfüllt:

$$\int_{-\infty}^{\infty} f(x)\delta(x)dx = f(0) \tag{2.7}$$

für jede Testfunktion $f : \mathbb{X} \rightarrow \mathbb{R}$

Das Ziel des Partikelfilters als ein Bayes-Filter ist es, die Partikelmenge so weiterzuentwickeln, dass sie die Posterior-Verteilung des nächsten Zeitpunkts $p_{k+1}(x_{k+1})$ unter Berücksichtigung des Prozessmodells, des Beobachtungsmodells und der Beobachtung z_{k+1}.repräsentiert.

Der grundlegendste und häufigste Implementierungsrahmen des Partikelfilters ist der sequenzielle Wichtigkeitssampling- und Resampling-Filter. Er besteht aus grundlegenden Schritten, die einen Iterationszyklus bilden:

Wichtigkeitssampling

Eine Voraussetzung für die Umsetzung und auch für die Ableitung des Partikelfilters ist die Fähigkeit, beliebige Wahrscheinlichkeitsdichtefunktionen (PDFs) mit Partikeln und Gewichten darzustellen. Dies ist durch die Methode des Importance Sampling möglich.
Das Ziel des Partikelfilters ist es, mit Hilfe der Partikel und Gewichte eine Wahrscheinlichkeitsdichtefunktion p darzustellen, die jedoch in der Regel unbekannt ist. Daher muss eine ähnliche Verteilung q, eine sogenannte Vorschlagsdichtefunktion, entworfen werden, aus der Partikel gezogen werden können.

$$p(x) > 0 \implies q(x) > 0 \quad \forall x \in \mathbb{R}^{n_x} \tag{2.8}$$

Die Wahrscheinlichkeitsdichtefunktion q wird als Vorschlagsdichte bezeichnet. In der Regel wird für q eine gleichmäßige oder gaußsche Verteilung gewählt. Wenn ν Partikel aus q gezogen werden, müssen deren nicht normalisierte Gewichte angepasst werden, um p darzustellen.

$$\widetilde{w}^{(i)} = \frac{p(x^{(i)})}{q(x^{(i)})} \tag{2.9}$$

und die normalisierten Gewichte ist

$$w^{(i)} = \frac{\widetilde{w}^{(i)}}{\sum_{j=1}^{\nu} \widetilde{w}^{(j)}} \tag{2.10}$$

Sequenzielles Wichtigkeitssampling

Die allgemeine Implementierung des Partikelfilters wird durch den sequenziellen Wichtigkeitssampling-Ansatz realisiert. Es wird angenommen, dass die posteriori Zustandsverteilung $p_k(x_k)$ des Zeitschritts k durch die Partikelmenge $\{x_k^{(i)}, w_k^{(i)}\}_{i=1}^{\nu}$ gemäß Gleichung (2.6) dargestellt wird.

Eine Beobachtung z_{k+1} tritt im Zeitschritt $k+1$ auf. Gegeben sind außerdem die Zustandsübergangswahrscheinlichkeit $f_+(x_{k+1}|x_k)$ und die Messwahrscheinlichkeit $g_{k+1}(z_{k+1}|x_{k+1})$, bekannt als Prozessmodell und Messmodell. Der erste Schritt des Partikelfilters zieht ν neue Partikel. Jedes neue Partikel $x_{k+1}^{(i)}$ wird gezogen, indem die Vorschlagsdichte $q_{k+1}(x_{k+1}^{(i)}|x_k^{(i)}, z_{k+1})$ verwendet wird, wobei die Konditionierung auf den entsprechenden Partikelzustand des letzten Zeitschritts $x_k^{(i)}$ und der Beobachtung z_{k+1} erfolgt.

Man kann zeigen, dass die posteriori Zustandsverteilung $p_{k+1}(x_{k+1})$ zum Zeitpunkt $k+1$ durch die neu gezogenen Partikel dargestellt wird, wenn die entsprechenden aktualisierten, nicht normalisierten Gewichte folgende Bedingung erfüllen:

$$\widetilde{w}_{k+1}^{(i)} = w_k^{(i)} \frac{g_{k+1}(z_{k+1}|x_{k+1}^{(i)}) f_+(x_{k+1}^{(i)}|x_k^{(i)})}{q_{k+1}(x_{k+1}^{(i)}|x_k^{(i)}, z_{k+1})} \tag{2.11}$$

und die normalisierten Partikelgewichte sind:

$$w_{k+1}^{(i)} = \frac{\widetilde{w}_{k+1}^{(i)}}{\sum_{j=1}^{\nu} \widetilde{w}_{k+1}^{(j)}} \tag{2.12}$$

Die Herleitung findet sich in [1].

Resampling

Wenn die sequenzielle Wichtigkeitssampling-Methode mehrmals ausgeführt wird, degenerieren die Partikel. Das bedeutet, dass sich das Gewicht auf eine kleine Anzahl von Partikeln konzentriert, während die meisten Partikel Gewichte nahe null tragen. Dies kann durch einen Resampling-Schritt vermieden werden. Im Resampling-Schritt werden ν Partikel aus einer Menge von ν Partikeln gezogen. Ein Partikel kann dabei mehrmals gezogen werden oder gar nicht, sodass Duplikate in der neuen Partikelmenge möglich sind. Die Wahrscheinlichkeit, dass ein Partikel ausgewählt wird, ist proportional zu seinem Gewicht. Nach dem Resampling-Schritt erhalten alle Partikel das gleiche Gewicht von $\frac{1}{\nu}$.

Sequenzielles Wichtigkeits-Resampling

Verschiedene Implementierungen des Partikelfilters können durch die angewandte Vorschlagsdichte $q_{k+1}(x_{k+1}^{(i)}|x_k^{(i)}, z_{k+1})$ unterschieden werden. Eine der am häufigsten verwendeten Methoden ist die Methode des sequenziellen Wichtigkeits-Resampling (Sequential Importance Resampling, SIR). In diesem Fall wird die Vorschlagsdichte wie folgt gewählt:

$$q_{k+1}\left(x_{k+1}^{(i)}\middle|x_k^{(i)}, z_{k+1}\right) = f_+(x_{k+1}^{(i)}|x_k^{(i)}) \tag{2.13}$$

Dies ist zwar nicht die optimale Wahl in Bezug auf die Varianz der Partikelgewichte, bietet jedoch praktische Vorteile. Aus den Gleichungen (2.11) und (2.12) folgt, dass der Aktualisierungsschritt des Partikelfilters durch Neuberechnung der Partikelgewichte durchgeführt wird:

$$\widetilde{w}_{k+1}^{(i)} = w_k^{(i)} g_{k+1}(z_{k+1}|x_{k+1}^{(i)}) \tag{2.14}$$

und anschließend normalisiert wird. Das SIR-Verfahren kann den Resampling-Schritt nur nach dem Aktualisierungsschritt ausführen. [1]

2.4 Dempster-Shafer-Theorie

2.4.1 Rahmen der Unterscheidung

Dies definiert eine Menge, die alle möglichen Hypothesen oder Zustände eines Systems umfasst. Im Tracker RFS wird dieses Rahmenwerk verwendet, um die Belegung und den freien Zustand eines Gitters darzustellen.

$$\Omega = \{O, F\} \tag{2.15}$$

O bedeutet die Belegung einer Gitterzelle. F bedeutet die Freiheit einer Gitterzelle.[1]

2.4.2 Grundlegende Wahrscheinlichkeitszuweisung

Die Basiswahrscheinlichkeitszuweisungsfunktion (Basic Probability Assignment, BPA) m ordnet jedem Teilmengenrahmen einen Wert im Bereich von [0, 1] zu, der das Vertrauen in diese Teilmenge darstellt.

BPA erfüllt zwei Bedingungen: $m(\emptyset) = 0$ und $\sum m(A) = 1$, wobei A eine beliebige Teilmenge des Rahmens ist.

- $m(O)$ steht für das Vertrauen, dass die Zelle belegt ist.
- $m(F)$ steht für das Vertrauen, dass die Zelle frei ist.
- $m(\Omega)$ steht für das Vertrauen, dass der Zustand der Zelle ungewiss oder widersprüchlich ist.

Diese BPA-Werte müssen die folgenden Bedingungen erfüllen:

- $m(O) + m(F) + m(\Omega) = 1$

2.4.3 Vertrauensfunktion und Plausibilitätsfunktion

Die Vertrauensfunktion (Belief Function, Bel) stellt das minimale Vertrauen in einen bestimmten Sachverhalt dar, also den Grad, in die Beweise diesen Sachverhalt eindeutig unterstützen. Sie wird durch die Summe der BPA-Werte berechnet.

Die Plausibilitätsfunktion (Plausibility Function, Pl) stellt das maximale Vertrauen in einen bestimmten Sachverhalt dar, also den Grad, in die Beweise diesem Sachverhalt nicht widersprechen. Sie wird berechnet, indem die BPA-Werte aller Teilmengen, die eine Schnittmenge mit dem Sachverhalt haben, aufsummiert werden.

Beispiel

Im Folgenden werden diese beiden Konzepte anhand eines Beispiels erklärt. Angenommen, ein Sensor liefert folgende Informationen. Für eine bestimmte Zelle gilt:

- $m(O) = 0{,}6$, was bedeutet, dass mit 60 % Vertrauen angenommen wird, dass die Zelle belegt ist.
- $m(F) = 0{,}3$, was bedeutet, dass mit 30 % Vertrauen angenommen wird, dass die Zelle frei ist.
- $m(\Omega) = 0{,}1$, was bedeutet, dass eine Unsicherheit von 10 % besteht.

Vertrauensfunktion:

- $Bel(O)$ stellt das minimale Vertrauen dar, dass die Zelle belegt ist. Da nur O selbst eindeutig die Belegung der Zelle unterstützt, gilt $Bel(O) = m(O) = 0{,}6$
- $Bel(F)$ stellt das minimale Vertrauen dar, dass die Zelle frei ist. Da nur F selbst eindeutig die Freiheit der Zelle unterstützt, gilt: $Bel(F) = m(F) = 0{,}3$
- $Bel(\Omega)$ stellt das Vertrauen in alle möglichen Zustände dar und ist immer gleich 1.

Plausibilitätsfunktion:

- $Pl(O)$ stellt das maximale Vertrauen dar, dass die Zelle belegt ist. Alle Teilmengen, die eine Schnittmenge mit O haben, sind $\{O, \Omega\}$, daher gilt:

  $$Pl(O) = m(O) + m(\Omega) = 0{,}6 + 0{,}1 = 0{,}7$$

- $Pl(F)$ stellt das maximale Vertrauen dar, dass die Zelle frei ist. Alle Teilmengen, die eine Schnittmenge mit F haben, sind $\{F, \Omega\}$, daher gilt:

  $$Pl(F) = m(F) + m(\Omega) = 0{,}3 + 0{,}1 = 0{,}4$$

Für die Belegung der Zelle ist ein minimales Vertrauen von 60 % und ein maximales Vertrauen von 70 %. Für die Freiheit der Zelle ist ein minimales Vertrauen von 30 % und ein maximales Vertrauen von 40 %. Diese Darstellungsweise ermöglicht, nicht nur das eindeutige Vertrauen (Bel) zu berücksichtigen, sondern auch die Möglichkeit (Pl). Sie ist besonders nützlich, wenn man mit Unsicherheiten und widersprüchlichen Informationen umgeht. [1][6]

3 Rahmenbedingungen

Nach der Berücksichtigung der mathematischen Grundlagen werden in diesem Kapitel die Rahmenbedingungen für das eigene Konzept im Detail erörtert. Sie umfassen die Auswahl der Softwareumgebung, die Konfiguration der Messtechnik, die Auswahl der Testumgebung sowie die Beschreibung des konkreten Versuchsablaufs.

In dieser Forschung wird das Problem der Mehrzielverfolgung unter Verwendung von LiDAR-Sensoren untersucht. Im Vergleich zu anderen Sensoren wie Kameras oder Radarsystemen bietet LiDAR eine höhere Präzision bei der Erfassung von Entfernungen und der Umweltstruktur. Da extreme Wetterbedingungen die Leistung der LiDAR-Sensoren erheblich beeinträchtigen können, wird es angenommen, dass in der Umgebung keine extremen Wetterbedingungen herrschen. Zudem wird bei der Bewegungsmodellierung der Ziele eine gewisse Annahme über die Konstanz der Geschwindigkeit der verfolgten Objekte getroffen.

Eine fundierte und umfangreiche experimentelle Umgebung ist fähig, die Anforderung vom Algorithmus bei der Anpassung der Parameter zu erfüllen. Deshalb ist es notwendig, die Experimente sinnvoll zu gestalten. Gängige Verkehrsszenarien sollten entworfen werden, um die Anwendbarkeit des Algorithmus zu überprüfen. Darüber hinaus werden für einige extreme klassische Situationen, wie die Verdeckung von Zielen, ebenfalls entsprechende experimentelle Simulationen durchgeführt. Durch die Bereitstellung reichhaltiger experimenteller Messdaten soll die Allgemeingültigkeit des Algorithmus erhöht werden. Das bedeutet, je umfangreicher die Messdaten sind, desto wahrscheinlicher ist es, ein Algorithmus zu entwickeln, die robusten und präzisen Ergebnisse der Verfolgung liefert.

3.1 Softwareumgebung

MATLAB (Matrix Laboratory) ist eine von MathWorks entwickelte fortgeschrittene Programmierumgebung, die weit verbreitet in den Bereichen numerische Berechnungen, Datenanalyse, Entwicklung des Algorithmus, Modellierung und Simulation Anwendung findet. In MATLAB können leistungsstarke Matrix- und Vektoroperationen durchgeführt werden. Viele Operationen lassen sich in MATLAB mit einer einfachen Syntax ausführen, was es besonders für die Verarbeitung großer Datenmengen und numerische Berechnungen geeignet macht. Darüber hinaus unterstützt MATLAB sowohl 2D- als auch 3D-Darstellungen, wodurch qualitativ hochwertige Grafiken und Diagramme erzeugt werden können. MATLAB stellt eine Vielzahl von Plot-Funktionen zur Verfügung, die sowohl für die Datenvisualisierung als auch für die Präsentation von Simulationsergebnissen verwendet werden können. Am wichtigsten ist, dass MATLAB ausreichende Toolboxes bietet, wie etwa für die Verarbeitung von LiDAR-Punktwolken und die Verfolgung von Zielen. Durch die Nutzung integrierter Beispiele lassen sich die enthaltenen Funktionen effizient aufrufen.

Im Rahmen der Implementierung dieses Algorithmus wurden die folgenden Toolboxes und zusätzliche Funktionen verwendet:

- `Lidar Toolbox (Version 2.3)` [11]
- `Sensor Fusion and Tracking Toolbox (Version 2.5)` [12]
- `Parallel Computing Toolbox (Version 7.8)` [13]
- `Automated Driving Toolbox (Version 5.5)` [14][15]
- `App Designer (Version 9.4)` [16]

3.2 Messdatenerfassungsgerät

Im Experiment zur Mehrzielverfolgung wurden drei verschiedene LiDAR-Sensoren, der Ouster OS1-64, der Leishen LS02 und der Livox Horizon, zur Datenerfassung ausgewählt. Die Ziele von der Verwendung drei LiDAR-Sensoren bestehen in der Verbesserung der Systemleistung, der Robustheit sowie der Abdeckung vielfältiger experimenteller Anforderungen.

Verschiedene LiDAR-Sensoren weisen unterschiedliche technische Parameter auf, wie etwa Erfassungsreichweite, vertikaler/horizontaler Sichtwinkel (FOV), Auflösung und Abtastrate. Durch die Auswahl mehrerer Sensoren können reichhaltigere Punktwolkendaten aus verschiedenen Blickwinkeln und Höhen erfasst werden. Besonders bei der genauen Modellierung von Zielen in komplexen Verkehrsumgebungen trägt der Einsatz mehrerer unterschiedlicher LiDAR-Sensoren zur Ergänzung der Auflösungen bei. Hochauflösende Sensoren erfassen detaillierte Nahbereichsziele, während Sensoren mit geringerer Auflösung und größerer Reichweite umfassendere Umgebungsinformationen liefern.

LiDAR-Sensoren verschiedener Hersteller oder Modelle unterscheiden sich in Leistung, Preis und Anwendungsbereichen. Die Auswahl verschiedener LiDAR-Sensoren im Experiment ermöglicht es, deren jeweilige Vor- und Nachteile sowie ihre Leistung bei der Mehrzielverfolgung zu evaluieren: Durch die Experimente können die Sensoren unter verschiedenen Straßenbedingungen, Lichtverhältnissen und Wetterbedingungen verglichen werden, um ihre Stärken und Schwächen zu identifizieren. Einige LiDARs eignen sich möglicherweise für hochpräzise Kurzstreckenanwendungen wie Parken oder städtische Straßen, während andere für die Fernwahrnehmung auf Autobahnen besser geeignet sind. Der Vergleich der verschiedenen Sensoren im Experiment ermöglicht es, für jede Anwendung den am besten geeigneter Sensor auszuwählen und die Anpassungsfähigkeit des Verfolgungsalgorithmus an verschiedene Verkehrsszenarien zu verbessern. Im Folgenden werden die technischen Parameter der drei Sensoren vorgestellt und verglichen.

3.2.1 Vergleich der Sensoren

Die folgende Erstellung der Tabelle mit den Parametern der drei Sensoren basiert auf den Referenzen [7][8][9][10].

Scanning

Tabelle 3.1 Parameter der Scanning der drei Sensoren

Parameter	Ouster	Livox	Leishen
Scanning-Technologie	mechanisch (rotierend)	Halbfestkörper (Flächenstrahl-Technologie)	Hybrid-Festkörper (rotierend)
Scanning-Muster	Linearscanning	Nicht-repetitives Scanning	Linearscanning
Kanäle	64		128

Die Tabelle 3.1 befasst sich mit den Parametern von der Scanning der drei Sensoren. Eine Rotationstechnologie vom Scanning kommt beim Ouester und beim Leishen zum Einsatz. Der Ouster besitzt ein traditionelles mechanisch rotierendes LiDAR-System mit 64 Laserstrahlen. Durch das rotierende Abtasten ermöglicht es eine vollständige 360°-Erfassung der Umgebung, wobei die Punktwolke gleichmäßig und kontinuierlich erfasst wird. Im Vergleich dazu basiert der Leishen auf einer hybriden Festkörpertechnologie, die MEMS-Spiegel mit teilweiser mechanischer Rotation kombiniert. Diese Technologie dient zum erweiterten Sichtfeld sowie größerer Erfassungsreichweite bei hoher Stabilität. Mit 128 Laserstrahlen und einer hohen vertikalen Auflösung ist dieser Sensor besonders geeignet für die Erstellung detaillierter 3D-Punktwolken. Der Livox Horizon hingegen ist ein halbfestes LiDAR-System, das den Einsatz mechanischer Komponenten minimiert und dadurch die Lebensdauer des Geräts erhöht. Durch die innovative, nicht-repetitive Scanning-Technologie wird eine hohe Abdeckung der Punktwolken erreicht. Bei jedem Scan werden unterschiedliche Punkte im Raum erfasst, wodurch nach mehreren Durchläufen eine dichte Punktwolke entsteht, die besonders detaillierte Umgebungsbilder verwirklicht.

Der Horizon verwendet sowohl die nicht-repetitive Scanning-Technologie als auch die Multi-Laser- und Multi-APD-DL-Pack-Technologie, was zu einer etwa dreimal höheren Punktdichte im Vergleich zur Livox Mid-Serie führt. Im Zeitverlauf erhöht sich die Abdeckung des Sichtfelds (FOV) signifikant, was detailliertere Umgebungsinformationen liefert. Zur Verdeutlichung der Unterschiede sind in Bild 3.1 die Scanning-Muster von 3 Sensoren gegenüberstellt. Abbildung (a) zeigt, wie die nicht-repetitive Scanning-Methode vom Livox die Abdeckung des Sichtfelds schrittweise erweitert. Im Gegensatz dazu bleibt in Abbildung 3.1(b) die Abdeckung konstant, da hier jeder Scan repetitiv erfolgt; was der Scanning-Methode von Ouster und Leishen entspricht.

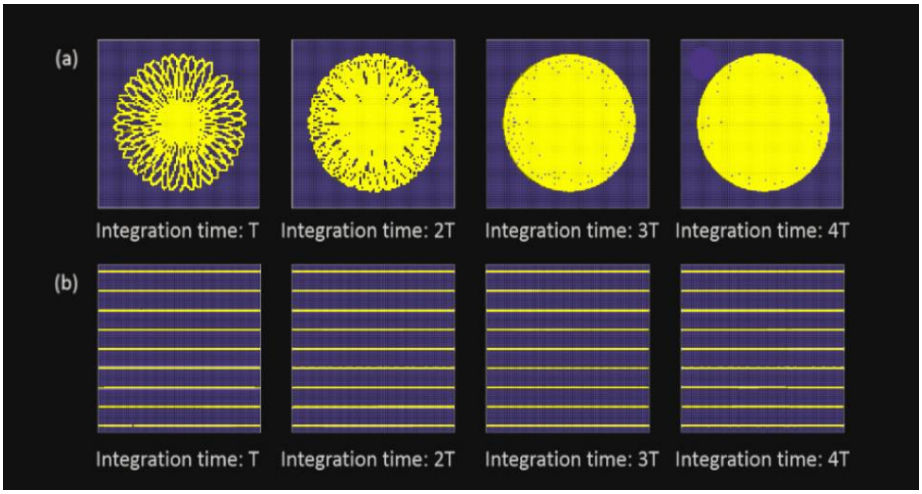


Abbildung 3.1 Vergleich der Scanning-Muster [7]

Die entsprechende im Sichtfeld innerhalb von 0,1 Sekunden Punktwolkenverteilung der Horizon befindet sich in Abbildung 3.2. Im Zentrum des Sichtfeldes ist die Scandichte höher, mit einem durchschnittlichen Linienabstand von 0,2°, was deutlich dichter ist als bei herkömmlichen 64-Linien-LiDAR-Sensoren, bei denen der Linienabstand zwischen 0,3° und 0,6° liegt.

Die beiden kreisförmigen Bereiche an den Seiten haben eine geringere Scandichte, mit einem durchschnittlichen Linienabstand von 0,4° (die meisten Linienabstände liegen zwischen 0,2° und 0,8°), was mit herkömmlichen 64-Linien-LiDAR-Sensoren innerhalb von 0,1 Sekunden vergleichbar ist.

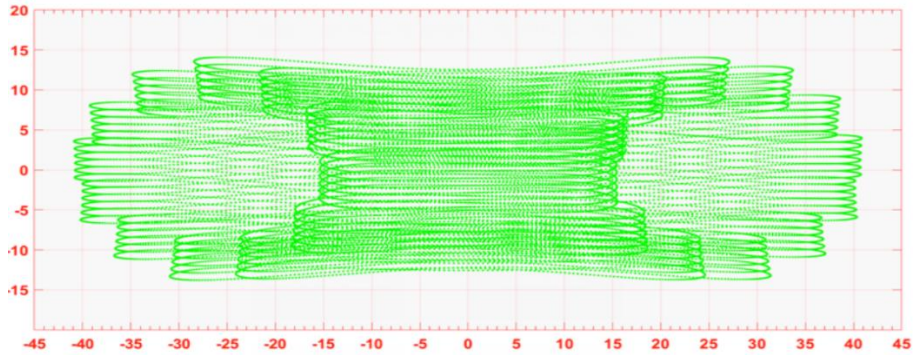


Abbildung 3.2 Punktwolkenverteilung des Livox Horizon während 0,1s [7]

Aus der Abbildung 3.3 werden die Abdeckungskurven des Sichtfeldes von verschiedenen Sensoren verglichen. Bei einer Integrationszeit von weniger als 0,1 Sekunden erreicht der Wert der FOV-Abdeckung des Horizon etwa 60 %, ähnlich wie bei einem 64-Linien-LiDAR-Sensor. Mit zunehmender Integrationszeit auf 0,5 Sekunden nähert sich seine FOV-Abdeckung aber 100 %. Das bedeutet, dass fast alle Bereiche von Laserstrahlen erfasst werden. [7]

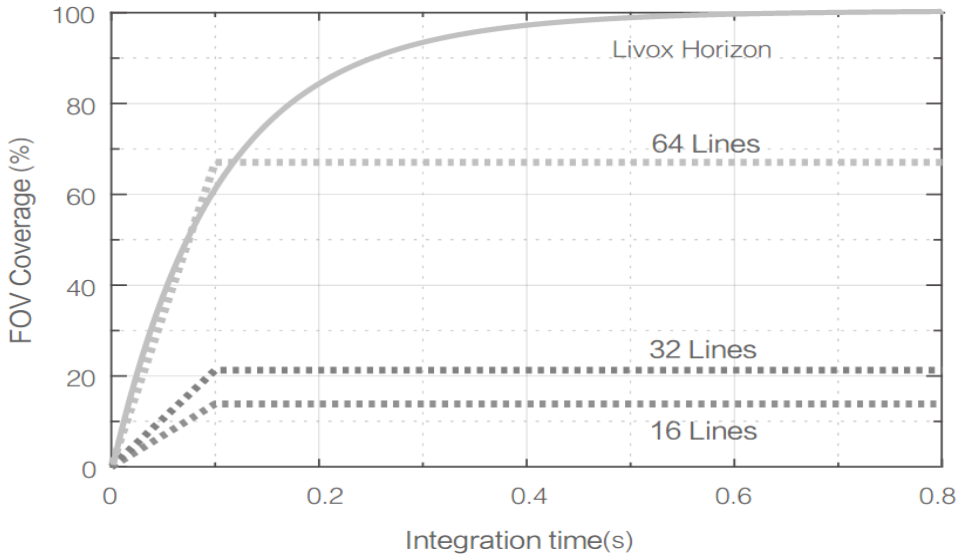


Abbildung 3.3 Vergleich der FOV-Abdeckung des LIVOX Horizon mit anderen Sensoren [7]

Wellenlänge des Lasers und Detektionsbereich

Tabelle 3.2 Wellenlänge des Lasers und Detektionsbereich der drei Sensoren

Parameter	Ouster	Livox	Leishen
Wellenlänge	855 nm	905 nm	1550 nm
Lasersicherheit	Klasse 1	Klasse 1	Klasse 1
Strahlaufweitung	0,13° (FWHM)	0,28° (Horiziontal) 0,03 (Vertikal)	
Erfassungsbereich (80% Reflektivität, 100 klx)	110 m @ >90% 150 m @ >50%	260 m	
Erfassungsbereich (10% Reflektivität, 100 klx)	50 m @ >90% 65 m @ >50%	90 m	250 m

Die mit der Welle zusammenhängenden Parameter von 3 Sensoren sind in Tabelle 3.2 aufgeführt. Der Leishen verfügt über die längste Wellenlänge, die 1550 nm beiträgt. Dann folgt der Livox mit 905 nm. Die Wellenlänge des Ouster liegt bei 855 nm. Dieser wesentliche Parameter ruft unterschiedliche Umweltbeständigkeit und Erfassungsbereiche hervor. Der Laser mit einer Wellenlänge von 855 nm unterliegt in der Luft einer deutlich erhöhten Dämpfung im Vergleich zu einem 1550-nm-Laser, was die maximale Erfassungsreichweite einschränkt. Bei einer Reflektivität von 10 % kann der Ouster bis zu 50 Meter erfassen. Er weist zudem eine eingeschränkte Durchdringungsfähigkeit bei ungünstigen Umweltbedingungen wie Nebel und Regen auf.

Der 905-nm-Laser vom Livox bietet jedoch dennoch eine gute Erfassungsleistung im mittleren Entfernungsbereich. Bei einer Reflektivität von 10 % kann der Livox Horizon bis zu 90 Meter erfassen,

was ihn für Anwendungen im Nah- und Mittelbereich prädestiniert. Der 905-nm-Laser weist eine durchschnittliche Durchdringungsfähigkeit bei Nebel, Regen und anderen Umwelteinflüssen auf, bleibt jedoch hinter der Leistung des 1550-nm-Lasers zurück. Dennoch bietet der Livox Horizon unter normalen Bedingungen eine zuverlässige Umgebungswahrnehmung.

Eine bessere Leistung bei der Langstreckenerfassung bringt der 1550-nm-Laser mit sich. Bei einer Reflektivität von 10 % ist der Leishen in der Lage, Ziele in einer Entfernung von bis zu 250 Metern zu erfassen. Ebenso wie oben erwähnt, ist er bei widrigen Wetterbedingungen als überlegend betrachtet.

Obwohl die Augensicherheit aller drei Sensoren Klasse 1 ist, wird die Effizienz des Ouster und des Livox unvermeidlich eingeschränkt, um die Anforderung an die Augensicherheit zu erfüllen. Während der 1550-nm-Laser außerhalb des empfindlichen Spektralbereichs der menschlichen Augen liegt, liegen ihre Wellenlängen im nahen Infrarotbereich.

FOV

Tabelle 3.3 Sichtfeld der drei Sensoren

Parameter	Ouster	Livox	Leishen
Horizontal FOV	360°	81,7°	120°
Vertikal FOV	33,2° (-16,6° ~16,6°)	25,1° (-12,55° ~12,55°)	25° (-12,5° ~12,5°)

In der Tabelle 3.3 geht es um dem Sichtfeld der 3 Sensoren. In diesem Bereich steht der Ouster an erster Stelle. Wegen seines vollständigen 360°-Horizontalsichtfeld ist er besonders gut für großflächige Umgebungswahrnehmungsaufgaben wie autonome Fahrsysteme und Robotik geeignet. Mit einem vertikalen Sichtwinkel von 33,2° gewährleistet er eine umfassende Abdeckung. Der Leishen LS02 bietet einen horizontalen Sichtwinkel von 120° und einen vertikalen Winkel von 25°. Das Schlusslicht bildet der Livox, dessen Sichtfeld mit einem horizontalen Winkel von 81,7° und einem vertikalen Winkel von 25,1° entspricht, was ihn ideal für Wahrnehmungsanwendungen im mittleren bis nahen Entfernungsbereich macht.

Die Wahl des passenden LiDAR-Sensors sollte von den spezifischen Anforderungen der jeweiligen Anwendungssituationen abhängen. Der Ouster OS1-64 ist aufgrund seines vollständigen Panoramablicks besonders geeignet für umfangreiche Umgebungswahrnehmungen. Im Vergleich zum Ouster liegen die Schwerpunkte des Livox und Leishen auf die Erfassung der präzisen Daten in einem engeren Sichtfeld, insbesondere für Aufgaben im mittleren bis nahen Entfernungsbereich.

Abtastrate

Tabelle 3.4 Abtastrate der drei Sensoren

Parameter	Ouster	Livox	Leishen
Abtastrate	1.310.720 pts/s	480.000 pts/s (dual return)	1.600.000 pts/s

Die Tabelle 3.4 enthält statistische Angaben über die Abtastrate der drei Sensoren. Den Daten zufolge ist die Anzahl der per Sekunde erfassten Punktwolken vom Ouster und Leishen relativ ähnlich. Der Leishen LS02 bietet eine sehr hohe Punktdichte, was 1.600.000 pts/s beträgt. Diese unterstützt die Anwendungen des Leishen unter Bedingungen hoher Auflösung und detaillierter Daten, wie zum Beispiel präzise Umweltmodellierung und autonomes Fahren. Zwar eine geringere Punktdichte weist der Livox Horizon auf, liefert jedoch dank seiner nicht-repetitiven Scanning-Technologie immer noch zuverlässige Daten. Er ist gut geeignet für Anwendungen mit mittleren Präzisionsanforderungen und für den Einsatz in komplexen Umgebungen in Echtzeit.

Fazit

Für fortschrittliche Anwendungen im autonomen Fahren sind verschiedene LiDAR-Sensoren je nach spezifischen Anforderungen und Umgebungsbedingungen von entscheidender Bedeutung. Der Leishen LS02 ist aufgrund seiner hohen Reichweite, Präzision und Punktdichte eine ausgezeichnete Wahl für Anwendungen, die hochpräzises Scannen über große Distanzen erfordern. Insbesondere bei widrigen Wetterbedingungen und in komplexen Umgebungen zeigt er seine Stärken, dank der Wellenlänge von 1550 nm, die nicht nur eine verbesserte Leistung bietet, sondern auch eine hohe Lasersicherheit gewährleistet.

Der Ouster OS1-64 hingegen eignet sich mit seinem 360°-Rundumsichtfeld und der Wellenlänge von 855 nm gut für autonomes Fahren, Robotik und Kartierungsanwendungen. Allerdings ist seine Reichweite begrenzt, und seine Anpassungsfähigkeit an schlechte Wetterverhältnisse stellt eine Herausforderung dar.

Für kostengünstige und zuverlässige Wahrnehmungsaufgaben im Nah- bis Mitteldistanzbereich bietet der Livox Horizon eine robuste Lösung. Mit einer Wellenlänge von 905 nm ist er besonders für Kurzstreckenanwendungen wie Sicherheitsüberwachung und einfachere Formen des autonomen Fahrens geeignet. Allerdings zeigt auch dieser Sensor Schwächen bei extremen Wetterbedingungen. Sowohl der Livox Horizon als auch der Ouster OS1-64 erfordern besondere Vorsichtsmaßnahmen hinsichtlich der Lasersicherheit, insbesondere in dicht besiedelten oder öffentlichen Bereichen.

3.2.2 Geräteverbindung

Die Abbildung 3.5 veranschaulicht, wie die Messtechnik zur Datenerfassung an den vorgesehenen Positionen installiert wird. Das System ist mit drei Sensoren ausgestattet, von denen jeder über einen eigenen Hub mit dem System verbunden ist. Diese Hübe stellen nicht nur die notwendige Stromversorgung für die Sensoren bereit, sondern sorgen auch für eine stabile Datenübertragung. Um den kontinuierlichen Betrieb der Sensoren sicherzustellen, ist außerdem eine externe Spannungsquelle im System installiert, die die Sensoren mit Strom versorgt.

Über einen Netzwerkverteiler sind die drei Sensoren mit dem für die Datenerfassung und -verarbeitung zuständigen Messrechner verbunden. Das Design des Netzwerkverteilers ermöglicht nicht nur eine effiziente und synchronisierte Datenübertragung, sondern auch die gleichzeitige Steuerung mehrerer Sensoren. Dank dieser Architektur kann das System Daten von den drei Sensoren aus verschiedenen Blickwinkeln gleichzeitig erfassen und vergleichen.

Dieser kooperative Arbeitsmodus der Mehrfachsensoren verbessert die Messgenauigkeit und die Vollständigkeit der Daten erheblich. Durch den Vergleich der Messergebnisse aus den unterschiedlichen Perspektiven der Sensoren kann die Genauigkeit der Daten effektiver überprüft werden, was die wissenschaftliche Fundierung und Zuverlässigkeit der Experimente weiter steigert. Dies bildet eine solide Datengrundlage für die nachfolgende Analyse, die Validierung von Algorithmen und deren Optimierung.

Abbildung 3.4 Strukturdiagramm von Geräteverbindung

3.3 Versuchsaufbau

Im ersten Teil dieses Abschnitts wird die Testumgebung eingehend untersucht, während im zweiten Teil die Abmessungen der Versuchsfahrzeugen darstellt werden. Abschließend wird der Versuchsablauf im Detail erläutert.

3.3.1 Testgelände

Um aussagekräftige Messdaten zu erhalten, muss das Testgelände bestimmte grundlegende Bedingungen erfüllen. Zunächst ist es erforderlich, dass die Testumgebung weitgehend frei von Störgeräuschen ist, wobei die Testszene klar definiert und umsetzbar bleiben sollte. Für das vorliegende Experiment wird das speziell ausgestattete Testgelände des K-Gebäudes der HTW Dresden genutzt. Ein wesentlicher Vorteil dieses Testgeländes ist das Fehlen anderer beweglicher Objekte, was ideale Voraussetzungen für hochpräzise Messungen schafft.

Darüber hinaus ist das Testgelände mit Fahrbahnmarkierungen und Straßenbegrenzungen ausgestattet, die die Durchführung wiederholbarer Teststrecken ermöglichen. Dies gewährleisteten die Zuverlässigkeit und Konsistenz der Experimente. Das Testgelände erlaubt nicht nur die Eliminierung externer Störungen, sondern bietet auch die Möglichkeit zur Durchführung präziser Kontrolltests, die optimale Bedingungen für die Erprobung und Validierung von Technologien wie Fahrzeuglokalisierung, Routenplanung und Multi-Target-Tracking schaffen.

Gleichzeitig ist die Umgebung so realitätsnah wie möglich gestaltet, um die Übertragbarkeit der Versuchsergebnisse auf reale Fahrsituationen zu gewährleisten. Daher stellt das Testgelände eine ideale Plattform zur Validierung von Multi-Target-Tracking-Algorithmen und deren Anwendung im autonomen Fahren dar.

Abbildung 3.5 Versuchsgeländekarten[17]

3.3.2 Versuchsfahrzeuge

In den Experimenten zur Verfolgung mehrerer Ziele werden Fahrzeuge unterschiedlicher Größe und Erscheinung berücksichtigt. Diese Fahrzeuge unterscheiden sich in ihren Formmerkmalen, Farben und Reflexionsraten. Die Verfolgungsalgorithmen müssen über eine gute Erkennungsfähigkeit verfügen, um diese Ziele in komplexen Szenarien präzise voneinander unterscheiden zu können. Fahrzeuge unterschiedlicher Größen und Erscheinungen stellen unterschiedliche Herausforderungen bei der Erkennung dar. Faktoren wie die Farbe, die Form und die Glattheit des Fahrzeugs können zu variierenden Reflexionseigenschaften auf den Sensoren führen, was die Effektivität der Erkennung beeinflusst.

Durch das Testen von Fahrzeugen unterschiedlicher Größe und Erscheinung kann sichergestellt werden, dass der Algorithmus sowohl eine robuste Erkennungsgenauigkeit als auch eine stabile Verfolgung bietet. Die Unterschiede in der Fahrzeuggröße beeinflussen die Häufigkeit und die Art der gegenseitigen Überdeckung der Ziele. Größere Fahrzeuge verdecken häufig kleinere Fahrzeuge oder lassen diese in visuellen oder LiDAR-Systemen verschwinden, was die Verfolgung erschwert.

Da die Größe eines Fahrzeugs direkten Einfluss auf dessen Verhalten im LiDAR hat, müssen bestimmte allgemeine Parameter des Verfolgungsalgorithmus an verschiedene Zielgrößen angepasst werden. Die Einführung von Fahrzeugen unterschiedlicher Größe kann dazu beitragen, die Parameter des Verfolgungsmodells zu optimieren, sodass es bei unterschiedlichen Fahrzeugtypen zuverlässig funktioniert.

Tabelle 3.5 liefert Informationen zu den Abmessungen und dem Erscheinungsbild der an den Experimenten beteiligten Fahrzeuge. Die Erstellung der Tabelle orientiert sich an den Referenzen [36][37][38][39]. Das Modell der Fahrzeuge befindet sich im Anhang 3.1.

Tabelle 3.5 Abmessungen der Versuchsfahrzeuge

ID	Automodell	Hauptfarbe	Abmessungen					
			l [m]	b [m]	h [m]	s [m]	V [m^3]	Maßstab [-]
1	Renault Twizy	orange	2,34	1,4	1,45	1,54	4,75	1
2	Passat GTE	weiß	4,88	1,83	1,48	2,71	13,22	2,8
3	BMW i3	weiß & schwarz	4,01	1,78	1,6	2,33	11,42	2,4
4	Porsche Cayenne	schwarz	4,92	1,98	1,7	2,78	16,56	3,5

3.4 Versuchsablauf

Bei der Entwicklung und Evaluierung von MTT-Algorithmen ist es notwendig, eine Vielzahl von realistischen Verkehrsszenarien zu testen. Denn unterschiedliche Verkehrsszenarien beinhalten eine Vielzahl komplexer Faktoren, wie die Anzahl der zufälligen Ziele und die große Variation in den Zieltypen, die die Leistung des Multi-Target Tracking-Algorithmus beeinflussen können. Der Algorithmus muss in der Lage sein, sich an die Eigenschaften verschiedener Ziele anzupassen, wie z.B. unterschiedliche Geschwindigkeiten, Größen und Typen von Fahrzeugen. Ein umfassendes Experiment kann zudem dazu beitragen, die Robustheit und Anpassungsfähigkeit des Algorithmus in verschiedenen komplexen Szenarien zu validieren, um sicherzustellen, dass der Algorithmus in vielfältigen realen Situationen stabil und präzise arbeitet. Außerdem unterstützt sie die Bewertung der Skalierbarkeit des Algorithmus und die Rechenleistung bei einer erhöhten Anzahl von Zielen. Darüber hinaus können verschiedene Szenarien nicht nur dabei helfen, Probleme zu identifizieren, sondern auch wertvolle Daten und Feedback für die Verbesserung des Algorithmus liefern.

Die Sensoren werden, wie in der Abbildung 3.9 dargestellt, in der unteren rechten Ecke des Testgeländes platziert. Die vier Testfahrzeuge und die drei Testpersonen beginnen ebenfalls an den Eingängen auf der rechten Seite des Testgeländes.

Die Planungen der Fahrtrouten von Fahrzeugen in verschiedenen Szenarien sind in Abbildung 3.7 und Abbildung 3.8 gegenüberstellt.

Abbildung 3.6 Versuchskarte von Szenario 01

Abbildung 3.7 Versuchskarte von Szenario 02

Im dritten Szenario, wie in Bild 3.9, fuhren zwei Autos nacheinander auf das Testgelände. Während das rote Auto den Spurwechsel abschloss, überholte das blaue Auto es. Der Twizy und der BMW i3 übernahmen dabei die Rolle des roten Autos in den beiden Versuchen. Der Passat überholte den Twizy. Entsprechend überholte der Porsche den BMW.

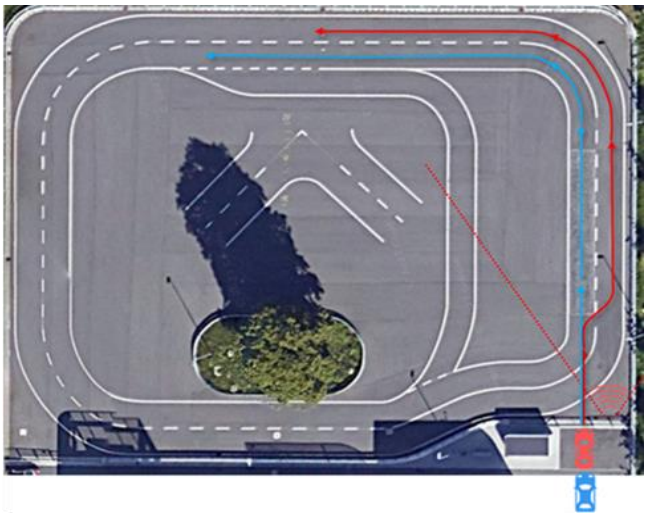
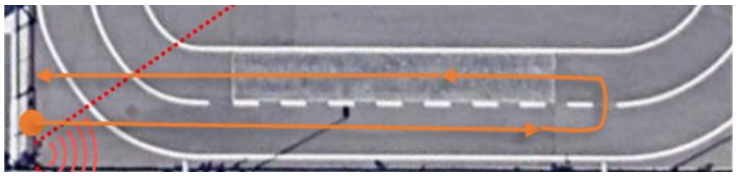
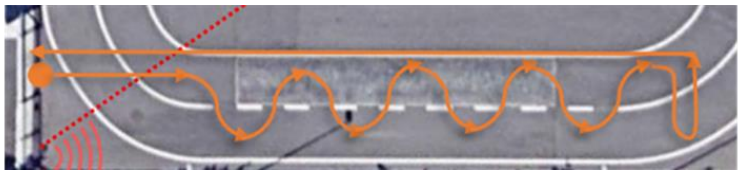


Abbildung 3.8 Versuchskarte von Szenario 03

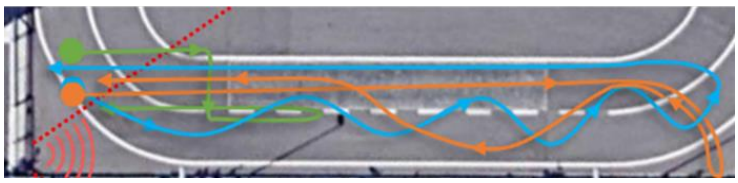
Bild 3.10 beschreibt das Routendesign der Fußgänger in unterschiedliche Szenarien. Im sechsten Szenario gab es drei Fußgänger. Fußgänger 1 folgte der blau markierten Strecke. Zunächst ging er eine S-Kurve und kehrte anschließend auf gerader Strecke zum Ausgangspunkt zurück. Fußgänger 2 folgte der orangen markierten Strecke. Er lief zunächst geradeaus, stoppte am Ende für kurze Zeit und kehrte dann zurück. Der Rückweg begann mit einer S-Kurve und führte dann auf gerader Strecke weiter. Fußgänger 3 folgte der grün markierten Strecke. Er lief zunächst eine Weile geradeaus, hielt dann an und ging zur Mittellinie der Straße. Dort lief er entlang der Mittellinie und stoppte anschließend. Danach kehrte er auf gerader Strecke zurück.



(a) Szenario 4



(b) Szenario 5



(c) Szenario 6

Abbildung 3.9 Versuchskarten in Szenarios 4, 5 und 6

4 Entwicklung einer Objektverfolgung

Dieses Kapitel vermittelt den Prozess der Entwicklung des MTT, der in Abbildung 4.1 dargestellt ist. Die Realisierung einer Zielverfolgung ist nur dann möglich, wenn zuverlässige Messdaten zur Verfügung stehen. Wenn man jedoch Punktwolken-Daten formell mit Matlab verarbeiten möchte, muss das ursprüngliche Punktwolken-Dateiformat ebenfalls umgewandelt werden. Die Details des Umwandlungsprozesses werden in der Arbeit nicht beschrieben.

Die Vorverarbeitung dieser Messdaten basiert auf der Anzahl der Punkte im ROI (Region of Interest), um die erste und letzte Bildsequenz der Rohdaten entsprechend zu kürzen. Damit beginnt die Verarbeitung der relevanten Daten in möglichst kurzer Zeit, was die verkürzte Berechnungszeit zur Folge hat. Einige experimentelle Messdaten fehlen, wie im Anhang 4.1 zusammengestellt.

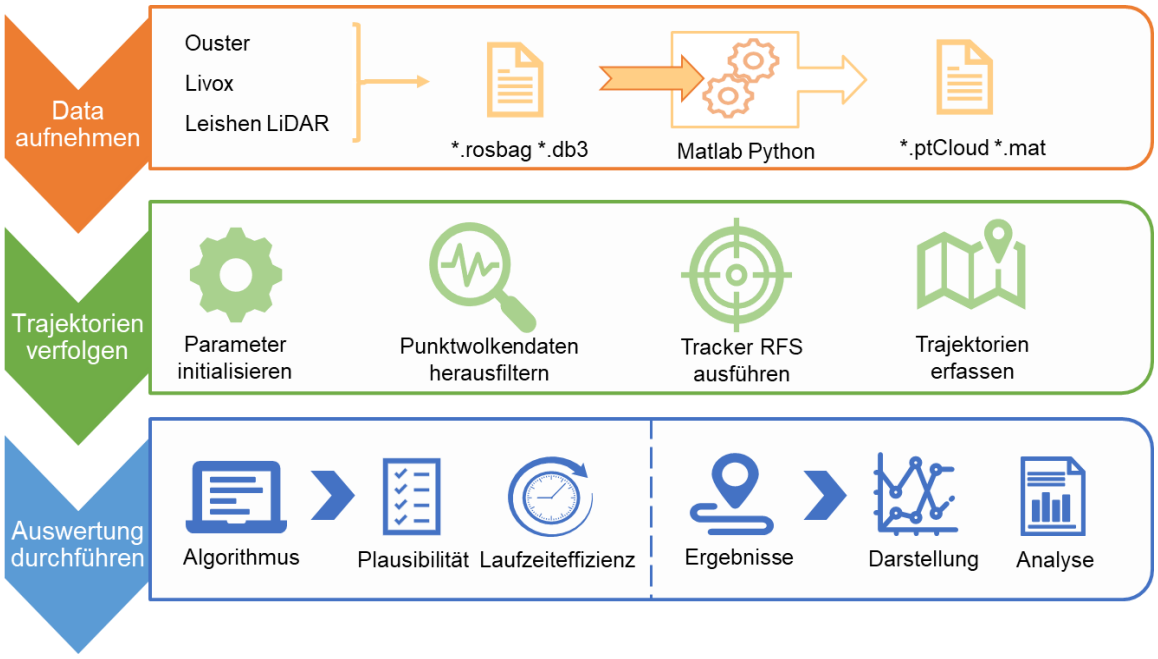


Abbildung 4.1 Prozess des Algorithmus

4.1 Zielverfolgung

Der Prozess der Zielverfolgung umfasst insgesamt fünf Schritte: das Einlesen der Punktwolkendaten, die Einstellung allgemeiner und spezifischer Parameter, die Ausführung des Trackers RFS sowie die Verarbeitung der Verfolgungsergebnisse.

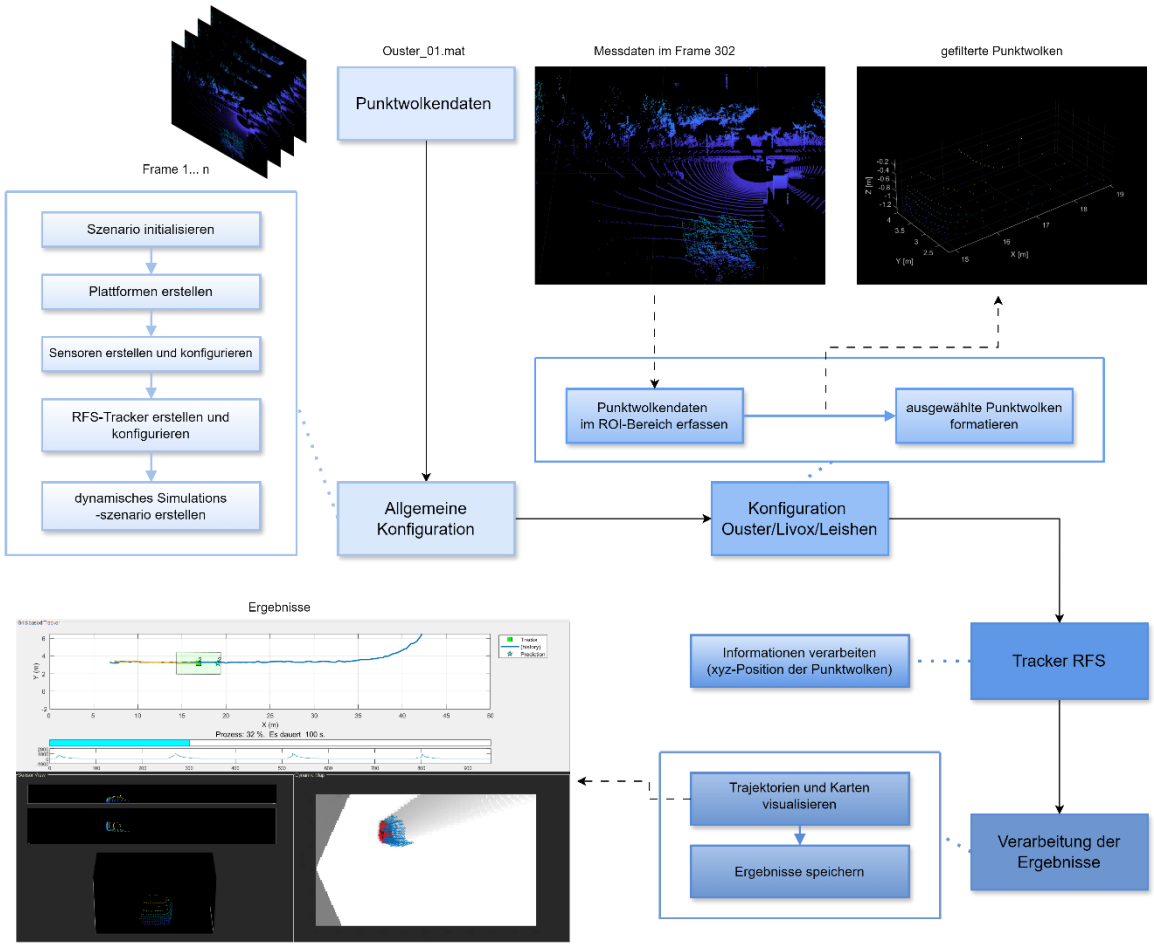


Abbildung 4.2 Ablaufdiagramm der Zielverfolgung

4.1.1 Allgemeine Konfiguration

Bei der allgemeinen Konfiguration werden die in Tabelle 4.1 aufgeführten Funktionen verwendet.

Tabelle 4.1 Funktionen von der allgemeinen Konfiguration

Funktion	Beschreibung
`trackingScenario`	Erstellt Verfolgungsszenario [22]
`platform`	Definiert ein Plattformobjekt, das zu einem Verfolgungsszenario gehört [23]
`trackingSensor Configuration`	Legen Sensorparameter wie Clustter-Dichte, Sensorgrenzen und Sensorauflösung fest [24]
`trackerGridRFS`	Ein Tracker, Erkennungen mehrerer Ziele von mehreren Sensoren in einer 2D-Umgebung zu verarbeiten [20]
`theaterPlot`	Darstellt Objekte, Erkennungen und Trajektorien im Szenario [25]
`trackPlotter`	Erstellt ein `TrackPlotter`-Objekt, das die Anzeige der Trajektorien in einem Vogelperspektiv-Diagramm konfiguriert [26]

4.1.2 ROI auswählen

Die Auswahl der Interessengebiete für die verschiedenen Sensoren basiert auf unterschiedlichen Koordinatenrichtungen und verschiedener Messmethoden, Wellenlängen der ausgesendeten Signale und Installationshöhen. Zudem muss es berücksichtigt werden, dass sich die Punktwolkenhöhe des verfolgten Objekts mit zunehmender Entfernung verändert. Dies kann dazu führen, dass ein festgelegter z-Bereich in der Nähe des Objekts möglicherweise nicht alle verfolgten Objekte erfasst, oder dass in der Ferne zu viel Rauschen detektiert wird, was die Genauigkeit der Erkennung beeinträchtigen kann. Daher ist ein fester z-Bereich nicht sinnvoll. Eine dynamische Anpassung je nach Entfernung ist erforderlich. Eine Möglichkeit besteht darin, zwei ROI-Bereiche zu verwenden, wobei der z-Bereich des nahegelegenen Bereichs insgesamt höher sein sollte als der z-Bereich des weiter entfernten Bereichs. Schließlich können die Daten der beiden ROI-Bereiche kombiniert werden, um eine ideale Punktwolke zu erhalten.

4.1.3 Projektion der Punktwolken-Daten

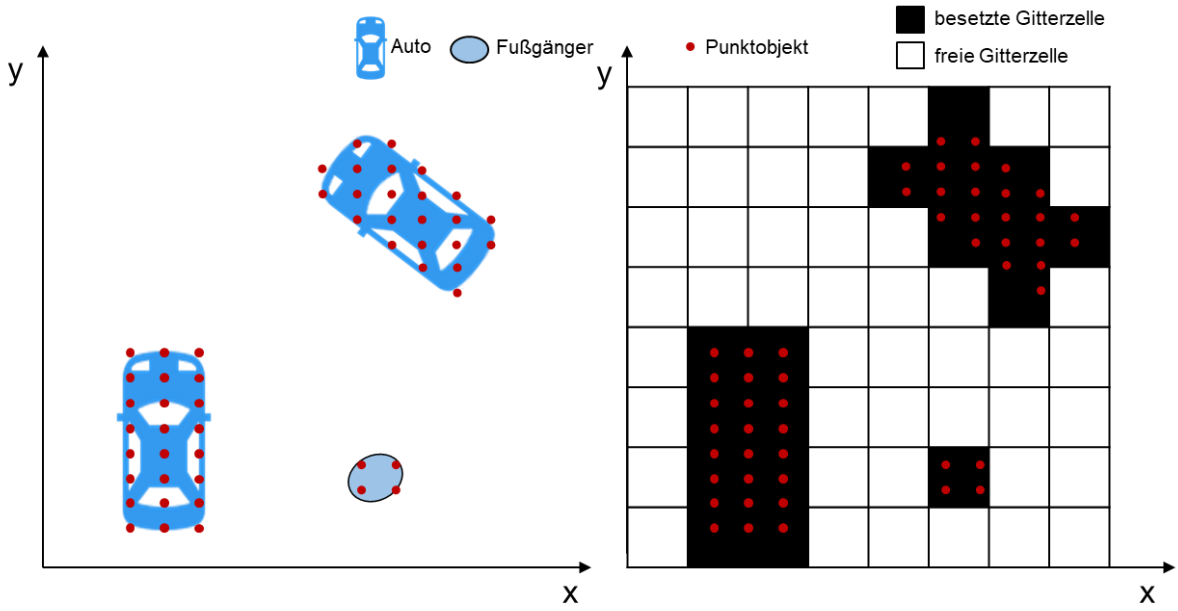


Abbildung 4.3 Diagramme über die Projektion der Punktwolken-Daten

Zu Beginn des Trackers wird die Punktwolkendaten zunächst auf ein 2D-Raster projiziert. Dies bildet die Grundlage für die anschließende Anwendung des Partikelfilters. Wie in Abbildung 4.3 dargestellt, repräsentieren die roten Punkte die erfassten Punktwolkendaten eines Objekts. In der Rastergrafik auf der rechten Seite werden die von Punktwolkendaten belegten Zellen schwarz dargestellt, während die nicht belegten Zellen weiß bleiben.

4.1.4 Umgang mit Ergebnissen

Um die Ergebnisse aus dem Tracker zu extrahieren und die Verfolgungsergebnisse anzuzeigen, müssen die relevanten Variablen identifiziert und ausgewertet werden. Diese Variablen enthalten Informationen wie die aktuelle Position, die Geschwindigkeit und die Historie sowie die Vorhersage der verfolgten Objekte. Danach visualisiert die Aktualisierung der Ergebnisse durch Funktionen `plotTrack`, `showdynamicMap` und `predictMapToTime`. Eine Übersicht von den Funktionen wird in Tabelle 4.2 dargestellt. Die Variablen stellen die Belegungssituation der Gitterzellen im Anhang 4.2 dar.

Tabelle 4.2 Verwendete Funktionen bei der Aktualisierung des Trackers

Funktion	**Beschreibung**
`getOccupancy`	Erhält die Belegungswahrscheinlichkeiten aller Gitterzellen in der Karte [27]
`getEvidences`	Erhält die geschätzten Belegungs- und Freiraumhinweise [28]
`getVelocity`	Erhält die Geschwindigkeitsschätzungen aller Gitterzellen in der Karte [29]
`getState`	Erhält den vollständigen geschätzten Zustand und die zugehörige Unsicherheit [30]
`predictTracks ToTime`	Vorhersagt des Zustands der Trajektorien [31]
`plotTrack`	Zeigt Objektspuren in einer Vogelperspektiv-darstellung an [32]
`showdynamicMap`	Zeichnet eine dynamische Belegungs-gitterkarte [33]
`predictMap ToTime`	Prognostiziert die dynamische Karte zu einem bestimmten Zeitstempel [34]

Die Variablen in Tabelle 4.3 werden als endgültige Verfolgungsergebnisse gespeichert. Die Strukturgrößen der Variablen in Tabelle 4.3 sind in Abbildung 4.4 dargestellt. Für `T_IDs`, `T_time`, `T_yaw` handelt es sich um eindimensionale Daten, sodass die zweite Dimension mit der ersten übereinstimmt und daher weggelassen werden kann. Bei `T_v` und `T_list` hingegen müssen die Koordinaten sowie die Geschwindigkeiten in x- und y-Richtung berücksichtigt werden, weshalb die zweite Dimension erforderlich ist.

Tabelle 4.3 Variablen über die Verfolgungsergebnisse

Variable	**Beschreibung**
`T_IDs`	Zeichnet die jeder Trajektorie zugeordnete IDs auf
`T_time`	Zeichnet die entsprechende globale Frame-Nummern auf, in denen jede Trajektorie erscheint.
`T_list`	Zeichnet die jeder Trajektorie zugeordnet Positionen auf
`T_v`	Zeichnet die jeder Trajektorie zugeordnete Geschwindigkeiten auf
`T_yaw`	Zeichnet die jeder Trajektorie zugeordnete Yaw-Winkel auf

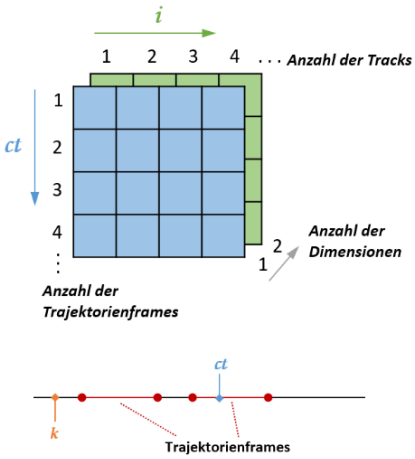


Abbildung 4.4 Darstellung der Größe der Variablen

4.2 begleitende App für die Mehrzielverfolgung

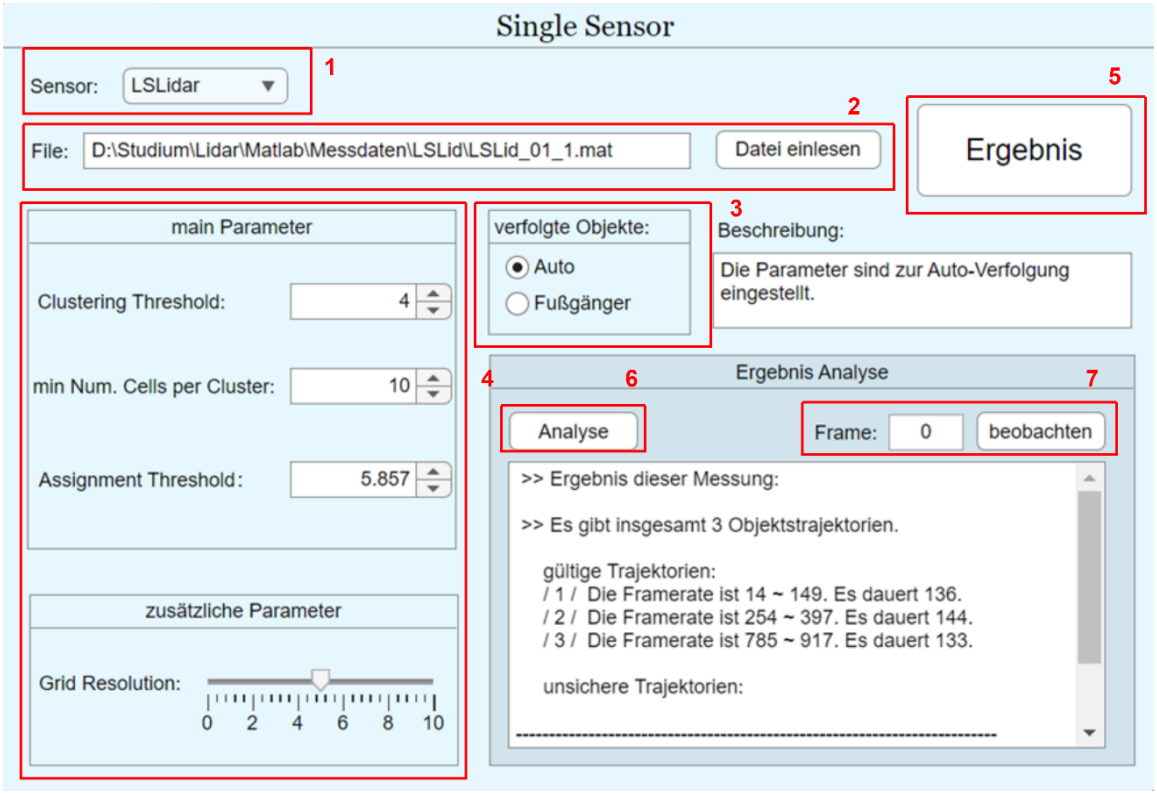


Abbildung 4.5 App-Oberfläche

In Bezug auf die im Abschnitt 4.1 beschriebene Implementierung des MOT wurde eine App entwickelt, die die Bedienung des MOT-Algorithmus vereinfacht. Die spezifischen Schritte zur Bedienung der App sind in der folgenden Abbildung 4.6 dargestellt.

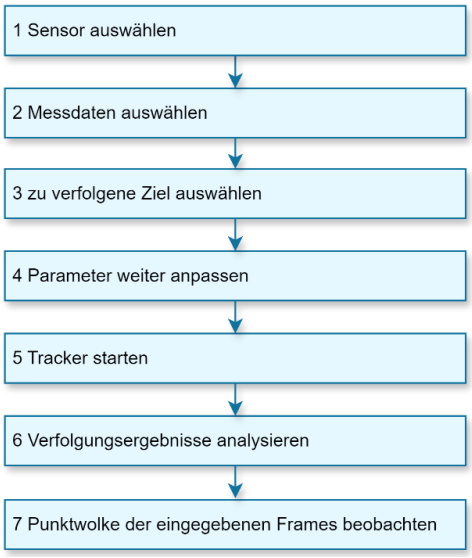


Abbildung 4.6 Bedienungsschritte

Durch das Ausführen der APP können die folgenden Ergebnisse erzielt werden.

Ergebnisanzeige

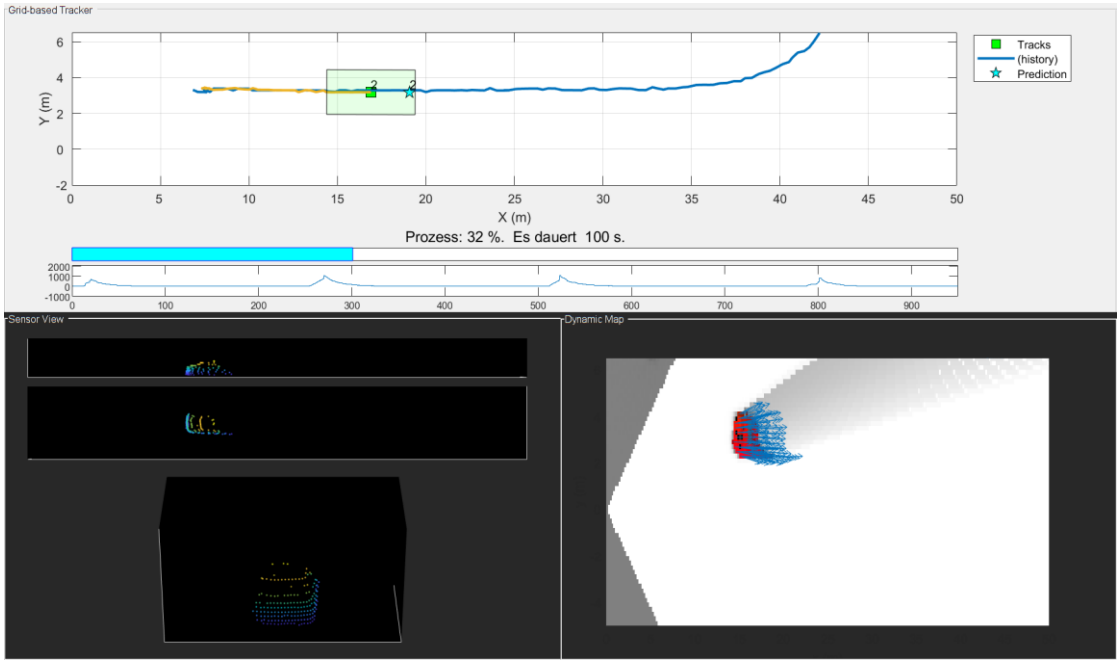


Abbildung 4.7 Ergebnisanzeige

Die obige Abbildung zeigt die Trajektorien der verfolgten Objekte. Die durchgezogenen Linien stellen die historischen Bewegungsverläufe dar, während die grünen Quadrate die aktuellen Positionen der Tracks markieren. Die Sterne repräsentieren die prognostizierten Positionen der Objekte zu einem zukünftigen Zeitpunkt.

Im mittleren Bereich befindet sich der Fortschritt der Verfolgung. Ein Liniendiagramm zeigt die Anzahl der Punkte innerhalb des festgelegten ROI an. In der unteren linken Ecke werden die Sensoren visualisiert, die zur Validierung der Verfolgung der Trajektorien verwendet werden. Diese Validierung trägt dazu bei, Fehl- oder Auslassungen von Objekten aufgrund ihrer Größe zu vermeiden. In der unteren rechten Ecke wird die dynamische Karte angezeigt, welche die Projektion der Punktwolken auf der Karte darstellt. Diese Karte bietet eine grobe Übersicht über den Status der erkannten Ziele. Im folgenden Kapitel wird die dynamische Karte im Detail erläutert.

5 Umsetzung des MTT mit Tracker RFS

In diesem Kapitel wird der Focus daraufgelegt, wie sich die Mehrzielverfolgung mit dem Tracker RFS realisiert. Die meisten MTT-Algorithmen stellen die Umgebung als eine Gruppe diskreter Objekte mit einer unbekannten Anzahl dar. Der Tracker RFS verknüpft die Sensormessungen direkt mit den objektbezogenen Hypothesen (Object-level hypothesis), die sich auf Annahmen oder Schätzungen über das Vorhandensein, den Zustand und die Eigenschaften jedes potenziellen Objekts beziehen. In erweiterten Ziel-Trackern sind sie deutlich komplexer als in Punktziel-Trackern, da jedes Objekt in den Sensordaten mehrere Messwerte erzeugen kann. Der gitterbasierte Tracker RFS gehört zu erweiterten Ziel-Trackern. Dabei ist ein komplexeres Beobachtungsmodell erforderlich.

Im folgenden Bild 5.1 ist ein Hauptprogrammablaufplan vom Tracker RFS aufgeführt. Sensormessdaten bilden die Grundlage für die Implementierung und Aktualisierung der Hypothesen. Nach dem Einlesen der Messdaten erfolgt die Durchführung der Partikelfilterung. Anschließend wird eine dynamische Belegungs-Gitterkarte als Zwischenrepräsentation der Umgebung erstellt. Auf dieser Grundlage wird das Management der Trajektorien vorgenommen.

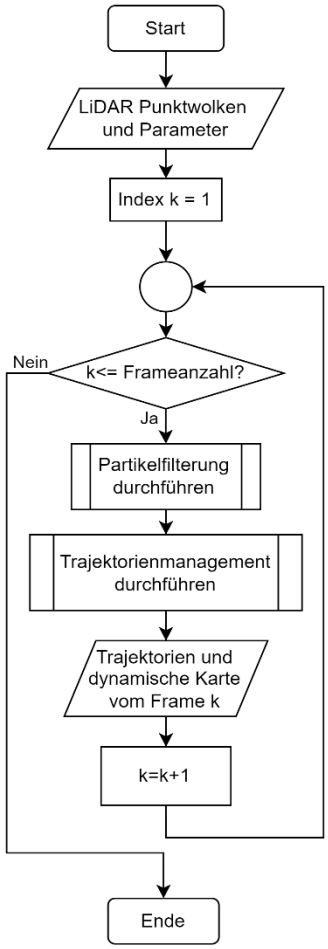


Abbildung 5.1 Hauptprogrammablaufplan vom Tracker RFS

5.1 Partikelfilterung

Eine partikelbasierte Vorgehensweise hat sich zur Ermittlung den Zustand jeder Gitterzelle etabliert. Sie aktualisiert mittels der bayesschen Methode die Zustandsschätzung dynamisch durch die Kombination von a priori Informationen und Beobachtungsdaten.

Zuerst wird die Geschwindigkeit der Partikel und andere unbekannte Zustände, wie Wendegeschwindigkeit und Beschleunigung, gleichmäßig anhand von durch a-priori-Beschränkungen bereitgestellten Vorinformationen gesampelt. Die Partikelverteilung der Vorhersage für den aktuellen Frame basierend auf dem vorherigen Frame ist bekannt. Unter Einbeziehung der aktuellen Beobachtungsdaten (Belegungsbeweis) werden die persistierenden Partikel im aktuellen Frame aktualisiert, um den Zustand aller gegenwärtigen Ziele anzupassen. Das bedeutet, dass die bayessche Methode eine a priori Verteilung, die das Systemverständnis vor den Beobachtungsdaten beschreibt, in eine a posteriori Verteilung überführt. Eine a posteriori Verteilung spiegelt den Systemzustand nach Einbeziehung der Beobachtungs-informationen wider.

Anschließend werden neue Partikel basierend auf der Geburtswahrscheinlichkeit erzeugt. Der Resampling-Schritt stellt sicher, dass die Anzahl der Partikel im Gitter konstant bleibt. Der Prozessablauf dieser Vorgehensweise wird in Abbildung 5.2 ausführlich erläutert.

Eine dynamische Belegungs-Gitterkarte kann als Zwischenrepräsentation der Umgebung verwendet werden, wobei die Umgebung in eine Reihe von zweidimensionalen Gitterzellen diskretisiert wird. Diese Karte visualisiert sowohl den Belegungszustand als auch die Kinematik des durch die Gitterzellen repräsentierten Raumes. In einem vorbereitenden Schritt wird die dynamische Karte dazu genutzt, die Zellen weiter in statische und dynamische Kategorien zu unterteilen, was die Herausfilterung von Messwerten statischer Objekte ermöglicht und die Rechenkomplexität verringert. Die relevanten Parameter von Partikelfilterung sind im Anhang 4.3 zusammengestellt.

Bewegungsmodell

In diese Experimente wird das Modell der konstanten Geschwindigkeit zur Zielverfolgung und Bewegungsprognose verwendet. Im Folgenden wird die Definition der Zustandsräume beschrieben.

Für das Modell der konstanten Geschwindigkeit sind die Zustandsvariablen des Partikels $[x;\ v_x;\ y;\ v_y]$ wobei x und y die Positionen in der x- bzw. y-Richtung darstellen und v_x sowie v_y die Geschwindigkeiten in diesen Richtungen sind. Der Zustand des Objekts umfasst zusätzlich den Gierwinkel sowie die Länge und Breitedes Objekts, dargestellt durch $[x;\ v_x;\ y;\ v_y; \mathrm{yaw}; \mathrm{L}; \mathrm{W}]$.

Ablauf

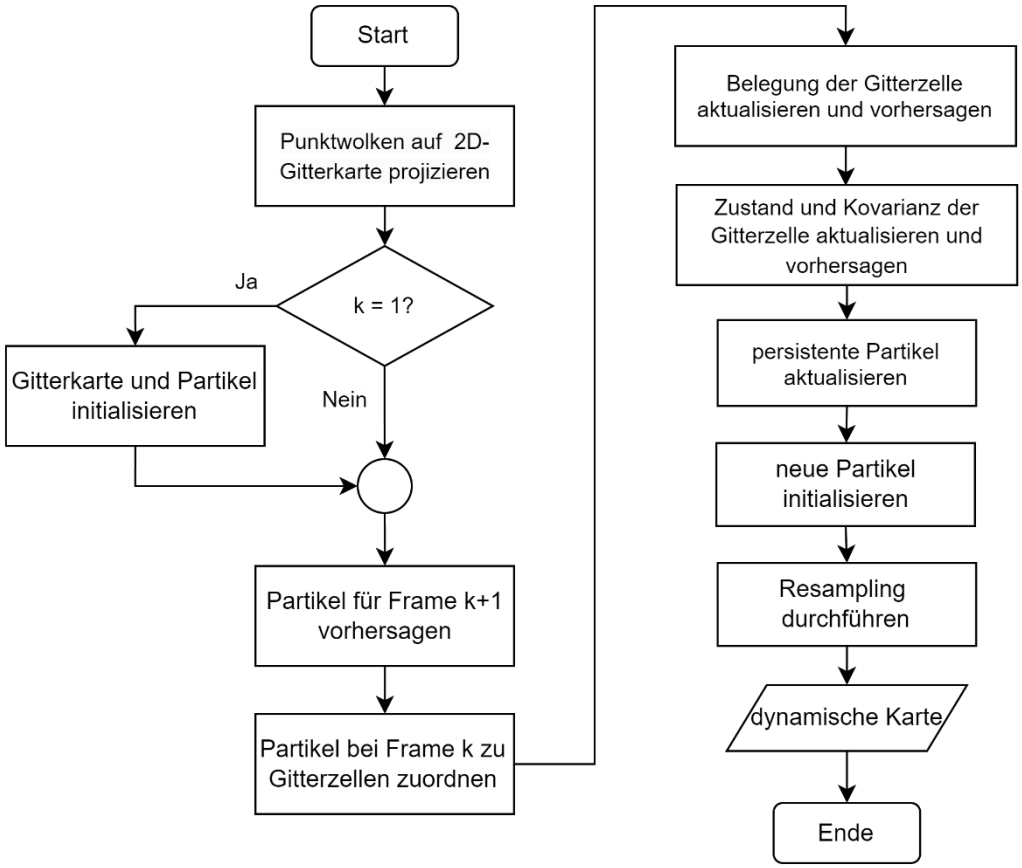


Abbildung 5.2 Programmablaufplan der Partikelfilterung

Initialisierung

Durch die Initialisierung der Parameter ist der Partikelfilterprozess mit der Messkarte abzustimmen. Diese Parameter entsprechen den Gittereigenschaften der Karte, sodass die räumliche Verteilung der Partikel die Merkmale der Zielumgebung widerspiegelt. Darüber hinaus werden Parameter für die Zustandsabschätzung des Verfolgungssystems zugewiesen, die die Positionsinformation in Bezug auf das Koordinatensystem steuern, um eine präzise Verfolgung der Zielposition zu gewährleisten. Partikel und ihre Gewichte werden anhand von Belegungsnachweisen initialisiert, damit die Partikel zu Beginn den Belegungszustand der Umgebung sinnvoll erfassen und so effektive a-priori-Informationen für den weiteren Filterprozess bereitstellen.

Vorhersage der Partikel

Im Rahmen der Partikelvorhersage erfolgt die Prognose durch das Zustandsübergangsmodell und das Prozessrauschen. Das Bewegungsmodell beschreibt, wie sich der Zustand des Ziels bei einer gegebenen Eingabe verändert. Aufgrund von Unsicherheiten in der Umgebung wird Rauschen

hinzugefügt, um eine breitere Abdeckung des Zustandsraums zu gewährleisten. Auf der Gitterkarte bedeutet dies, dass sich Partikel von einer Zelle zur nächsten bewegen können, manchmal auch über mehrere Zellen hinweg.

Bei der Vorhersage kontinuierlicher Ziele werden Partikel aus der Vorschlagsdichte q_{k+1} entnommen, um das vorhergesagte Ziel darzustellen. In den meisten Fällen wird die Übergangsdichte f_+ als Vorschlagsdichte verwendet, um den Sampling-Prozess zu vereinfachen:

$$q_{k+1}\left(x_{k+1}|x_k^{(i)}, Z_{k+1}\right) = f_+(x_{k+1}|x_k^{(i)}) \tag{5.1}$$

Der Vorhersage-Partikelsatz für den Zeitschritt k+1k+1k+1 wird als $\{x_{p,+}^{(i)}, w_{p,+}^{(i)}\}_{i=1}^{\nu}$ dargestellt, wobei die Gewichte der Partikel durch die Multiplikation mit der fortdauernden Wahrscheinlichkeit p_S aktualisiert werden,

$$w_{p,+}^{(i)} = p_S \cdot w_k^{(i)} \tag{5.2}$$

Dieser Partikelsatz beschreibt die vorhergesagte Wahrscheinlichkeitsdichtefunktion für fortdauernde Punktziele,

$$D_{p,+}(x_{k+1}) = \sum_{i=1}^{\nu} w_{p,+}^{(i)} \delta(x_{k+1} - x_{p,+}^{(i)}) \tag{5.3}$$

Zusammenfassend lässt sich sagen, dass der Vorhersage-Partikelsatz durch Sampling aus der Übergangsdichte und Anpassung der Gewichte basierend auf der fortdauernden Wahrscheinlichkeit generiert wird, um die PHD des kontinuierlichen Ziels im nächsten Zeitschritt zu modellieren.[1]

Darstellung und Aktualisierung des Gitterbesetzungszustands

Der Filter verwendet die grundlegende Glaubenszuweisung $m: 2^{\{O,F\}} \rightarrow [0,1]$, um den Besetzungszustand von Gitterzellen darzustellen. Daher speichert jede Gitterzelle die Besetzungsmasse $m(O)$ und die freie Masse $m(F)$.

Im Zeitabschnitt k zeigt der Filter den posterioren Zustand der einzelnen Gitterzelle c an, wobei die Partikelmenge $\{x_k^{(i,c)}, w_k^{(i,c)}\}_{i=1}^{\nu(c)}$. Hierbei repräsentiert die Summe der Partikelgewichte die Besetzungsmasse,

$$m_k^{(c)}(O_k) = \sum_{i=1}^{\nu(c)} w_k^{(i,c)} \tag{5.4}$$

Die Verteilung der Partikel stellt die räumliche Wahrscheinlichkeitsdichtefunktion (PDF) in der Gitterzelle c dar,

$$p_k^{(c)}(x_k) = \frac{1}{m_k^{(c)}(O_k)} \sum_{i=1}^{\nu(c)} w_k^{(i,c)} \delta\,(x_k - x_k^{(i,c)}) \tag{5.5}$$

Der Tracker aktualisiert die Existenzwahrscheinlichkeit von Punktzielen im Gitterzelle c unabhängig von der räumlichen Verteilung der Punktziele. Daher benötigt die Aktualisierung der Existenzwahrscheinlichkeit der Ziele die folgenden Informationen:

- Beobachtete Besetzungs-Grundglaubenszuweisung $m_{z_{k+1}}^{(c)}: 2^{\{O,F\}} \rightarrow [0,1]$
- Räumliche Likelihood-Funktion $g_{k+1}^{(c)}(z_{k+1}|x_{k+1})$
- Assoziationswahrscheinlichkeit $p_{A,k+1}^{(c)}$ zwischen der Likelihood-Funktion und den Punktzielen

Aktualisierung der fortlaufenden Partikel

Zunächst erfasst die Methode die Gewichte der aktuellen Partikel und berechnet die assoziierten Gewichte, indem sie diese mit den Gewichten der zugehörigen Partikel multipliziert. Es wird ein kumulativer Gewichtungsarray berechnet, um die Gesamtzahl der effektiven Partikel zu erhalten. Anschließend werden die Normalisierungsfaktoren für assoziierte und nicht-assoziierte Ziele getrennt berechnet, um die Gewichte der Partikel anzupassen. Basierend auf diesen Normalisierungsfaktoren werden die Gewichte jedes effektiven Partikels aktualisiert.

Partikelgewicht

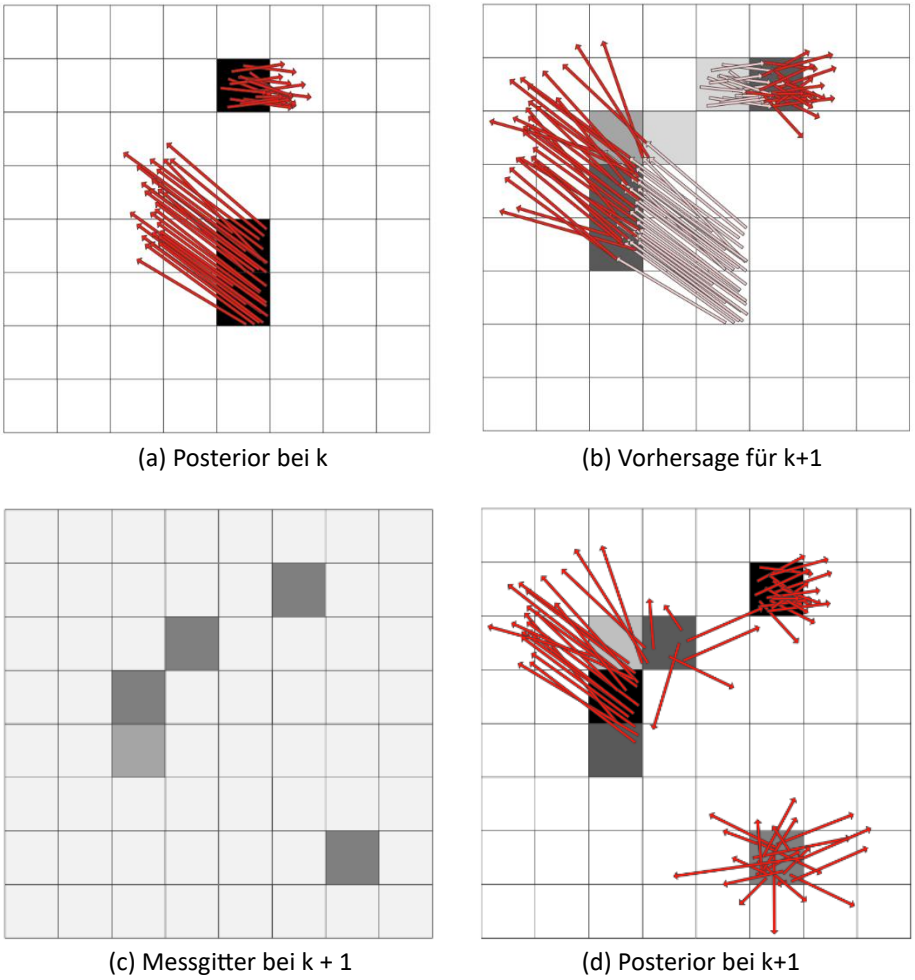
Partikel, die besser mit den Beobachtungen übereinstimmen, erhalten ein höheres Gewicht. Das Gewicht wird typischerweise durch den Abgleich der Sensordaten mit Informationen über Belegungsräume und Freiräume auf der Karte berechnet. Nach der Berechnung der Gewichte für jedes Partikel werden diese normalisiert, sodass die Summe der Partikelgewichte 1 ergibt. Dies stellt sicher, dass die Partikelgewichte als Wahrscheinlichkeiten interpretiert werden können.[1]

Resampling

Im Resampling werden bestehende Partikel mit neuen Partikeln kombiniert, um eine neue Partikelmenge zu bilden, während die entsprechenden Gewichte und Zellindizes beibehalten werden. Die neuen Gewichte werden basierend auf der Anzahl der Proben pro Zelle berechnet, und die Partikel, Gewichte und Zellindizes im Partikelfilter werden aktualisiert.

Mit der Zeit können einige Partikel sehr hohe Gewichte erhalten, während andere Gewichte nahe null haben. Um das Problem der Partikeldegeneration zu lösen, ist ein Resampling erforderlich. Beim Resampling werden Partikel mit hohen Gewichten beibehalten, während Partikel mit niedrigen

Gewichten verworfen werden. Neue Partikel werden in Bereichen hoher Wahrscheinlichkeit erzeugt, um sicherzustellen, dass die Verteilung der Partikel den aktuellen Zustand weiterhin korrekt repräsentiert. In der Gitterkarte bedeutet dies, dass resampelte Partikel sich auf bestimmte Zellen mit hoher Wahrscheinlichkeit konzentrieren, was darauf hindeutet, dass das verfolgte Ziel wahrscheinlich in diesen Zellen zu finden ist.



(a) Posterior bei k (b) Vorhersage für k+1

(c) Messgitter bei k + 1 (d) Posterior bei k+1

Abbildung 5.3 Verschiedene Zustände der rekursiven Schätzung der dynamischen Gitterkarte[1]

Abbildung 5.3 zeigt einen Teilprozess von der Partikelfilterung. (b) stellt die Vorhersage der Verteilung der Partikel zum Zeitpunkt k + 1 dar. Nach dem Erhalt der Messung der Belegungsnachweise zu dem Zeitpunkt k + 1, wie in (c) dargestellt, wird die Vorhersage korrigiert, was zu der Verteilung der Partikel zu dem Zeitpunkt k + 1 in Abbildung (d) führt.

5.2 Dynamische Karte

Dieser Filter berücksichtigt einen vierdimensionalen Raum, der aus einer zweidimensionalen Position und einer zweidimensionalen Geschwindigkeit besteht, und teilt den Raum in viele Gitterzellen auf. Eine Gitterzelle repräsentiert einen Bereich sowie die Geschwindigkeit in diesem Bereich.

Die vom Filter erzeugte Trajektorie wird auf einer dynamischen Gitterkarte überlagert. Die Farbe der dynamischen Gitterzellen wird gemäß dem Farbkreis definiert, der die Bewegungsrichtung im Szenenbild darstellt. Statische Zellen werden in einem Graustufenbild dargestellt, wobei der Grauton die Belegungswahrscheinlichkeit der Zellen anzeigt. Dynamische Zellen werden in einem RGB-Farbbild anhand von HSV-Werten angezeigt. [32]

Die folgende Tabelle 5.1 enthält eine detaillierte Beschreibung der verschiedenen Komponenten von HSV-Werten.

Tabelle 5.1 Darstellung der Begriffe über dynamische Karte

Begriff	**Bedeutung**
Farbton	Der Farbton wird durch den Richtungswinkel des Geschwindigkeitsvektors geteilt durch 360 berechnet. Wenn der Farbton von 0 auf 1 ansteigt, ändern sich die Farben in folgender Reihenfolge: Rot, Orange, Gelb, Grün, Cyan, Blau, Magenta und wieder zurück zu Rot.
Sättigung	Die Sättigung bezieht sich auf die Verteilung der Geschwindigkeit der Gitterzelle in Bezug auf die Mahalanobis-Distanz (d) zu null Geschwindigkeit. Gitterzellen mit d > 4 werden als vollständig gesättigt dargestellt.
Helligkeit	Die Helligkeit repräsentiert die Belegungswahrscheinlichkeit der Zelle.

Die Mahalanobis-Distanz ist ein Maß für den Abstand eines Punktes von der Mitte einer Verteilung unter Berücksichtigung der Korrelationen zwischen den Variablen.

Im Gegensatz zur euklidischen Distanz berücksichtigt die Mahalanobis-Distanz die Korrelationen zwischen den verschiedenen Geschwindigkeitskomponenten. Die Inverse der Kovarianzmatrix skaliert die verschiedenen Richtungen entsprechend, sodass die Distanzmessung genauer wird.

Wenn die Nullgeschwindigkeit als Referenz verwendet wird, kann die Mahalanobis-Distanz verwendet werden, um zu beurteilen, wie stark die Geschwindigkeit der Gitterzelle von der Ruhe (Nullgeschwindigkeit) abweicht. Je größer die Distanz, desto größer ist der Unterschied zwischen der Geschwindigkeit in der Zelle und der Nullgeschwindigkeit.

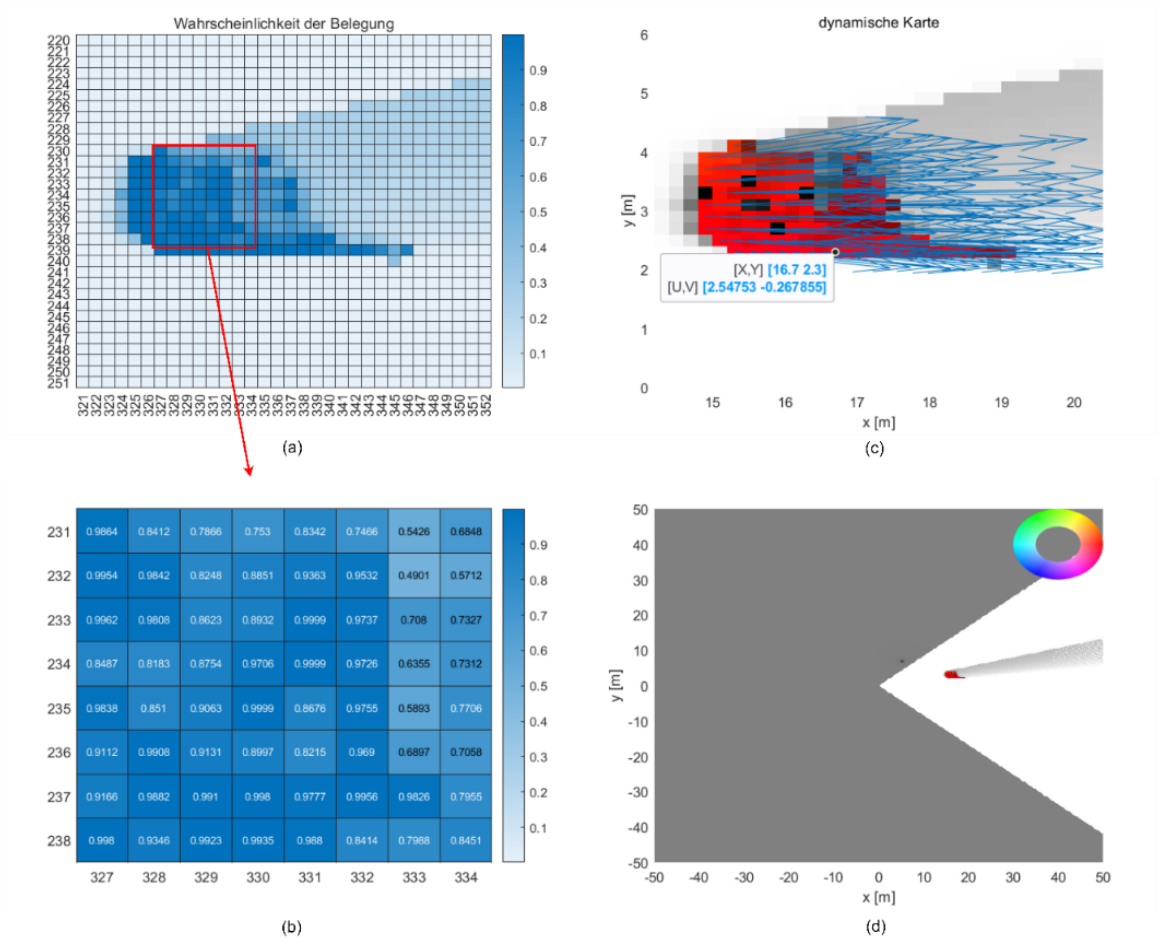
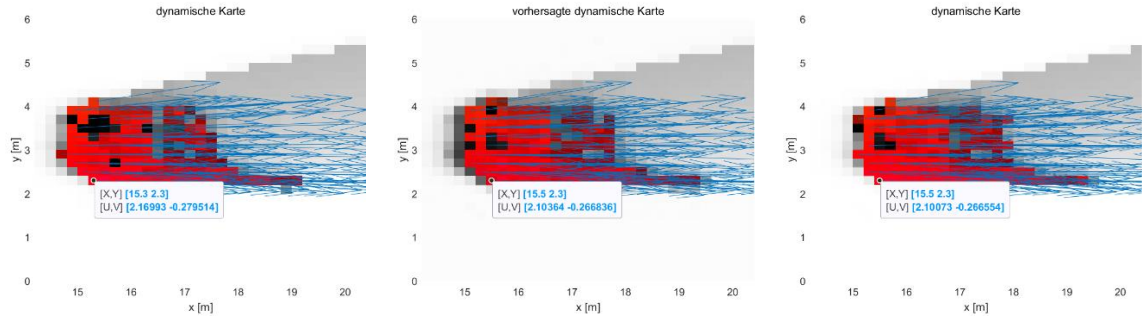


Abbildung 5.4 Belegungskarte und dynamische Karte zum gleichen Zeitpunkt

Abbildung 5.3(a) zeigt die Belegungswahrscheinlichkeitskarte, die nach der Verarbeitung der 2D-Gitterkarte der Punktwolke aus Frame 302 erstellt wurde. Je dunkler die Farbe, desto höher die Wahrscheinlichkeit, dass das Gitter belegt ist. Die spezifischen Belegungswahrscheinlichkeitswerte der Gitterzellen sind in Abbildung 5.3(b) dargestellt.

Abbildung 5.3(c) ist eine Vergrößerung von Abbildung 5.3(d). In Abbildung 5.3(d) ist erkennbar, dass sich das Fahrzeug annähernd entlang der x-Achse bewegt, was bedeutet, dass der Bewegungswinkel des Objekts etwa 0 Grad beträgt. Daher wird die Belegung der Gitterzellen des Objekts rot dargestellt. Abbildung 5.3(c) zeigt die spezifische Belegung der Gitterzellen durch das Objekt, einschließlich der Position der Gitterzellen, der entsprechenden x- und y-Geschwindigkeiten sowie der Richtung des Geschwindigkeitsvektors.



(a) dynamische Karte bei k (b) vorhersagte Karte für k+1 (c) dynamische Karte bei k+1

Abbildung 5.5 Dynamischen Karten zu verschiedenen Zeitpunkten sowie die Vorhersagekarten

Wie in der Abbildung dargestellt, hat sich die Vorhersage des von dem Auto besetzten Gitterelements in der linken unteren Ecke von (15,3, 2,3) zum Zeitpunkt k auf (15,5, 2,3) zum Zeitpunkt k + 1 verschoben, wobei die x-Geschwindigkeit 2,1 m/s und die y-Geschwindigkeit -0,27 m/s beträgt. Die vorhergesagte Position stimmt mit der in Abbildung (c) gezeigten überein, und der Geschwindigkeitsunterschied zwischen beiden ist ebenfalls gering. Daraus lässt sich schließen, dass die Abweichung zwischen dem Vorhersageergebnis und der Realität gering ist und die Vorhersage eine gewisse Zuverlässigkeit aufweist.

Im Vergleich zu einer objektbasierten Darstellung bietet eine gitterbasierte Darstellung Vorteile bei der Schätzung des dynamischen Zustands von Fahrzeugen. Dies bringt jedoch auch Einschränkungen mit sich. Rasterdarstellungen können das Konzept von Objekten nicht verstehen und daher nicht berücksichtigen, dass alle Teile eines Objekts konsistent bewegen.[1]

5.3 Verwaltung der Tracks

Im Falle der Erstellung einer dynamischen Karte können die dynamischen Gitterelemente weiterverarbeitet werden. In diesem Schritt wird parallele Berechnung eingesetzt, um 250.000 Gitterelemente gleichzeitig zu verwalten. Der Prozess des Managements der Trajektorien kann weiter unterteilt werden in die Initialisierung neuer Trajektorien, die Aktualisierung bestehender Trajektorien sowie das Löschen oder Bestimmen von unbestimmten Trajektorien, wie in Abbildung 5.7 dargestellt.

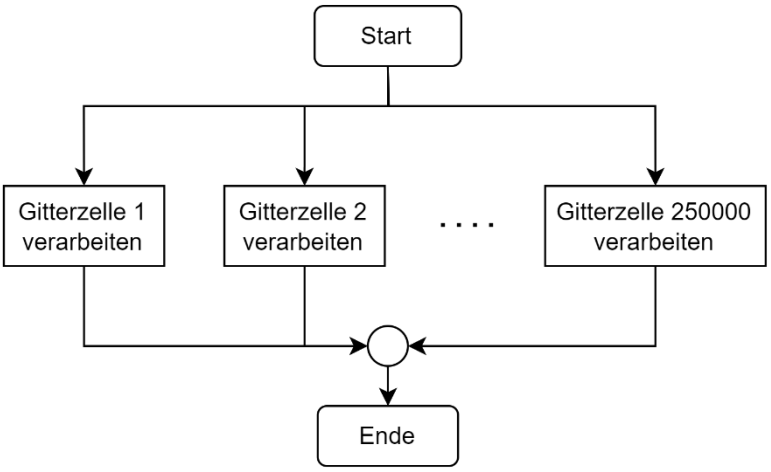


Abbildung 5.6 Parallele Berechnung der Gitterzellen

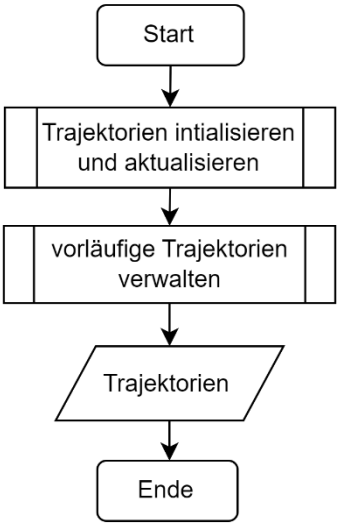


Abbildung 5.7 Hauptprogrammablaufplan der Verwaltung der Tracks

Abbildung 5.8 beschreibt detailliert den Prozess der Erstellung neuer Trajektorien und der Aktualisierung vorhandener Trajektorien. Die Initialisierung neuer Trajektorien basiert auf den in Tabelle 5.2 aufgeführten Parametern. Dazu verwendet der Tracker den DBSCAN-Algorithmus, um nicht zugewiesene dynamische Gitterzellen zu clustern.

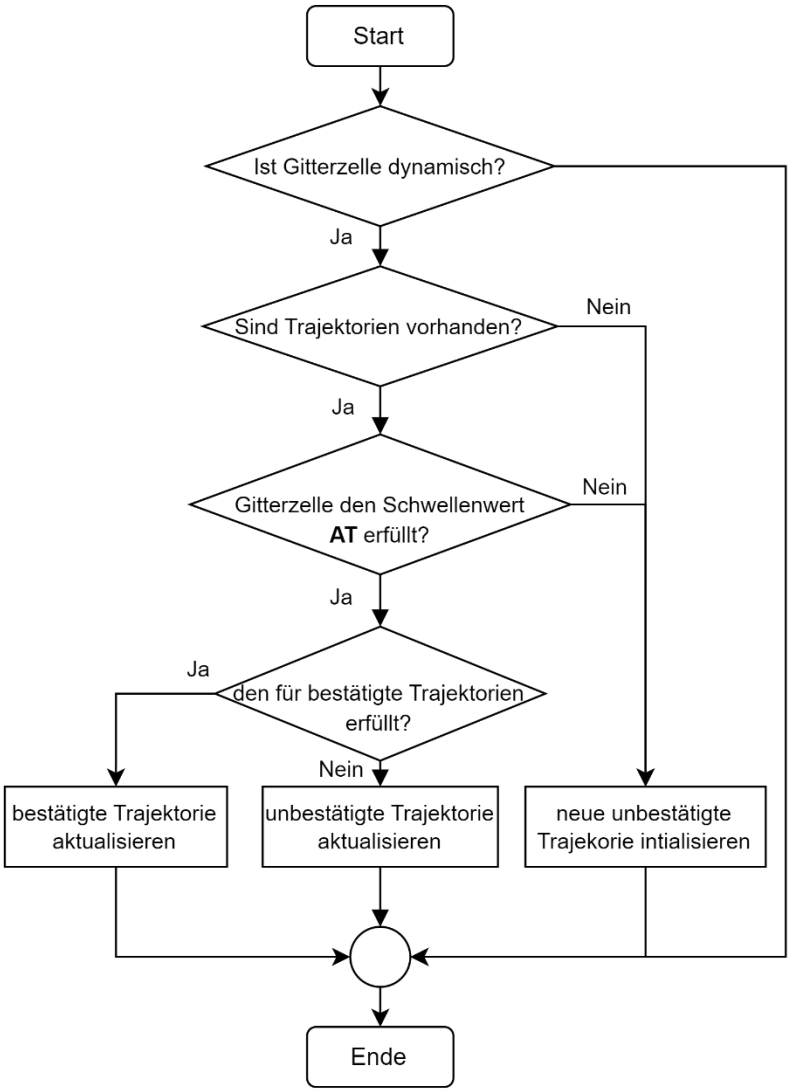


Abbildung 5.8 Programmablaufplan der Initialisierung und der Aktualisierung der Trajektorien

Tabelle 5.2 Parameter bei Initialisierung der Tracks

Parameter	Beschreibung	Initialwerte
`Clustering`	Wenn ‚DBSCAN' angegeben ist, wird das Clustering der nicht zugeordneten dynamischen Gitterzellen mit dem DBSCAN-Algorithmus durchgeführt.	‚DBSCAN'
`ClusteringThreshold`	Schwellenwert für das DBSCAN-Clustering	5
`MinNumCellsPerCluster`	Mindestanzahl von erforderlichen Punkten in einem Cluster	2

Bei der Aktualisierung der Tracks kann eine dynamische Gitterzelle nur dann mit einer Trajektorie assoziiert werden, wenn ihre Distanz zur Trajektorie kleiner ist als der zugewiesene Schwellenwert. Wenn eine dynamische Zelle nicht mit einer Trajektorie verbunden ist, mit der sie verbunden sein

sollte, muss der Schwellenwert AT erhöht werden. Umgekehrt sollte der Schwellenwert gesenkt werden, wenn eine dynamische Zelle mit einer Trajektorie verbunden ist, mit der sie nicht verbunden sein sollte.

Tabelle 5.3 Parameter bei Aktualisierung der Tracks

Parameter	Beschreibung	Initialwerte
`AssignmentThreshold`	Schwellenwert für die Zuordnung einer dynamischen Gitterzelle zu einer Spur. Eine dynamische Gitterzelle kann nur dann einer Spur zugeordnet werden, wenn der Abstand zur Spur kleiner ist als der `Assignment-Threshold`.	30

Der Prozess zur Bestätigung des Status vorübergehender Trajektorien sowie die benötigten Parameter sind in Abbildung 5.10 und Tabelle 5.3 dargestellt.

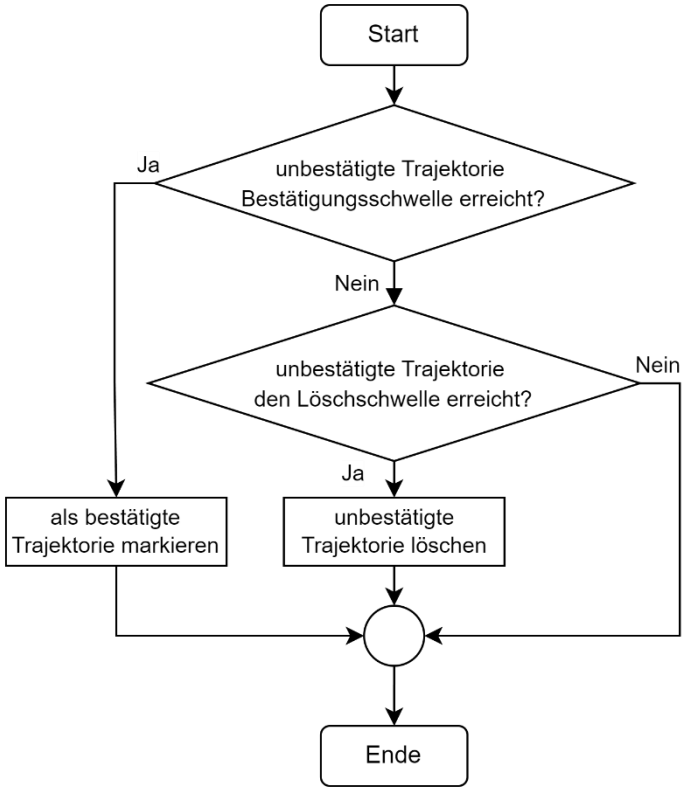


Abbildung 5.9 Programmablaufplan der der vorläufigen Tracks

Tabelle 5.4 Parameter bei Verwaltung der vorläufigen Tracks

Parameter	Beschreibung	Initialwerte
`ConfirmationThreshold`	Bestätigungsschwellenwert, wobei eine Spur bestätigt wird, wenn sie mindestens M von N Aktualisierungen erhält.	[2 3]
`DeletionThreshold`	Löschungsschwellenwert, wobei eine Spur gelöscht wird, wenn sie in den letzten R Aktualisierungen mindestens P Mal keiner dynamischen Gitterzelle zugewiesen wurde.	[5 5]

5.4 Fazit

Durch die Verteilung der Partikel auf einer Gitterkarte verarbeitet die Partikelfilterung effizienter bei großen Umgebungen. In weiterer Kombination mit der Gitterkarte ermöglicht sie, die Unsicherheit in diskreten Umgebungen zu behandeln. Außerdem lassen sich redundante Partikel durch Resampling bei der Gitterkarte effektiv, sodass die Partikel in Hochwahrscheinlichkeitsbereichen konzentriert sind. Die oben genannten Vorteile weisen darauf hin, warum der RFS-Tracker geeignet für die Zielverfolgung ist. Seiner Zielverfolgungsprozess ist ein Bottom-up-Prozess. Das heißt, seine Umsetzung erfolgt von Partikeln über Gitter hin zu Objekten.

Ein effektives Zusammenspiel von Track-Management und Zustandsabschätzung ist entscheidend für den Verfolgungsalgorithmus. Denn beide beeinflussen sich im Wesentlichen gegenseitig. Eine präzise Mehrzielzustandsabschätzung bildet die Grundlage für das Track-Management. Ist die Zustandsabschätzung der Ziele unpräzise, wird es schwierig, qualitativ hochwertige Trajektorien zu erzeugen. Umgekehrt können die Informationen über die Tracks in den Zielabschätzungsprozess zurückfließen und zur Verbesserung der Schätzungen beitragen.

6 Auswertung

Dieses Kapitel beschäftigt sich mit den Auswertungen des Algorithmus sowie der Verfolgungsergebnisse in vielerlei Hinsichten. Von dem Algorithmus und der Verfolgungs-ergebnisse wird die Auswertung ausgegangen.

Bei der Auswertung wird von dem Algorithmus und den Verfolgungsergebnissen ausgegangen. In Bezug auf das Algorithmus werden seine Genauigkeit und Effizienz analysiert. Nicht nur die Trajektorien, sondern auch Verhalten der Punktwolke und die Belegungswahrscheinlichkeit der Gitterzellen lassen bei der Bewertung von den Verfolgungsergebnisse sich berücksichtigen.

Für spezielle Verfolgungssituationen, wie z.B. verdeckte Objekte oder die Verschlechterung der Verfolgungsergebnisse durch den Stillstand des Objekts, werden ein zusätzlicher Algorithmus eingeführt. Mithilfe des Algorithmus ist es möglich, diese Situationen rechtzeitig zu erkennen.

6.1 Genauigkeit des Algorithmus

Generalized Optimal Sub-Pattern Assignment (GOSPA) wird in der Mehrzielverfolgung eingesetzt, weil es eine umfassendere Leistungsbewertung bietet als einfachere Metriken, die nur die Lokalisierungsgenauigkeit berücksichtigen.

GOSPA ermöglicht die Zerlegung des Gesamtfehlers in drei Komponenten:

- Lokalisierungsfehler (Fehler bei den Positionen zugewiesener Ziele)
- Fehler durch übersehene Ziele (Strafe für das Übersehen tatsächlicher Ziele)
- Fehler durch Fehlalarme (Strafe für falsche positive Erkennungen)

6.1.1 GOSPA Algorithmus

Zum Zeitpunkt t_k ist ein Tracker eine Menge der Tracks:

$$X = [x_1, x_2, \dots, x_m] \tag{6.1}$$

Zum Zeitpunkt t_k erhält eine Menge der Wahrheiten:

$$Y = [y_1, y_2, \dots, y_n] \tag{6.2}$$

Im Allgemeinen lautet die GOSPA-Metrik, einschließlich der Komponente (SGOSPA), wie folgt:

$$SGOSPA = (GOSPA^r + SC^r)^{1/r} \tag{6.3}$$

wobei r die Ordnung der Metrik, SC (switching component) die Umschaltkomponente, und

$GOSPA$ Basis-GOSPA-Metrik darstellen. Wenn $n \leq m$ gilt, lautet die GOSPA:

$$GOSPA = [\sum_{i=1}^{r} d_c^r(x_{\pi(i)}, y_i) + \frac{c^r}{\alpha}(m-n)]^{1/r} \quad (6.4)$$

wobei d_c die cutoff-basierte Distanz und $x_{\pi(i)}$ den Track repräsentieren, der der Wahrheit y_i zugewiesen wird. Die cutoff-basierte Distanz d_c ist definiert als:

$$d_c(x,y) = \min\{d_b(x,y), c\} \quad (6.5)$$

wobei c der Grenzwert der Distanzschwelle ist. $d_b(x,y)$ bedeutet die Basisdistanz zwischen dem Track x und der Wahrheit y, die durch die Distanzfunktion berechnet ist. Die cutoff-basierte Distanz d_c ist der kleinere Wert von d_b und c. α ist der Alpha-Parameter.

SC ist:

$$SC = SP \times n_s^{1/r} \quad (6.6)$$

wobei SP (switching penalty) die Umschaltstrafe ist und n_s die Anzahl der Umschaltungen darstellt. Wenn ein Track die Zuordnung von einer Wahrheit zu einer anderen wechselt, wird die Anzahl der Umschaltungen als 1 gezählt. Wenn ein Track von zugewiesen zu nicht zugewiesen wechselt oder von nicht zugewiesen zu zugewiesen, wird die Anzahl der Umschaltungen als 0,5 gezählt.

Wenn $\alpha = 2$ ist, kann die GOSPA-Metrik in drei Komponenten zerlegt werden:

$$GOSPA = [loc^r + miss^r + false^r]^{1/r} \quad (6.7)$$

Die Lokalisierungskomponente (loc) wird wie folgt berechnet:

$$loc = [\sum_{i=1}^{h} d_b^r(x_{\pi(i)}, y_i)]^{1/r} \quad (6.8)$$

wobei h die Anzahl der nicht trivialen Zuordnungen ist. Eine triviale Zuordnung ist, wenn ein Track keiner Wahrheit zugewiesen wird. Die Komponente der verpassten Ziele wird berechnet als:

$$miss = \frac{c}{2^{1/r}}(n_{miss})^{1/r} \quad (6.9)$$

wobei n_{miss} die Anzahl der verpassten Ziele darstellt. Die Komponente der falschen Tracks wird berechnet als:

$$false = \frac{c}{2^{1/r}}(n_{false})^{1/r} \quad (6.10)$$

wobei n_{false} die Anzahl der falschen Tracks ist. [35]

6.1.2 Beispiele

Da reale Daten nicht ausschließlich durch Sensoren erfasst werden können, ist es notwendig, Simulationen durchzuführen, um sowohl den Algorithmus als auch die Parameterkonfiguration zu bewerten. Dies stellt einen wichtigen Schritt zur quantitativen Analyse des Algorithmus dar. Im Simulationsszenario werden hauptsächlich Fahrzeuge mit normalen Abmessungen nachgebildet, während größere Fahrzeuge unberücksichtigt bleiben. Dies führt dazu, dass eine minimale Rasteranzahl von 10 ausreicht, um stabile Trajektorien zu generieren.

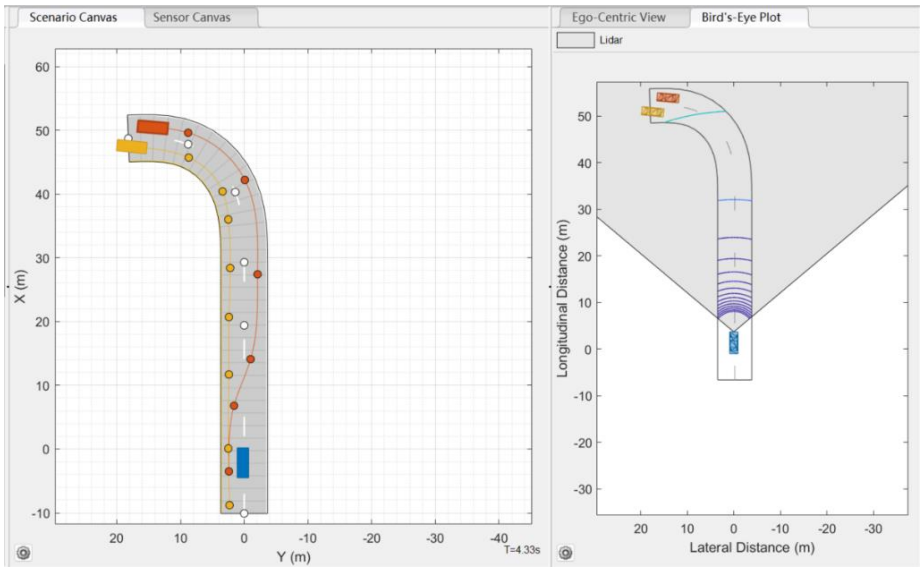


Abbildung 6.1 Simulation von Szenario 03

Die Bewertung verwendet die Funktion `trackGOSPAMetric`. In der Tabelle 6.1 werden deren Eingaben und Ausgaben vorgestellt.

Tabelle 6.1 Variablen über GOSPA

	Variablen	**Beschreibung**
Input	`tracks`	Verfolgungsinformationen
	`truths`	Ground-Truth-Daten
Output	GOSPA	GOSPA-Metrik
	`swithing`	Umschaltkomponente
	`localiyation`	Lokalisierungskomponente
	`missTarget`	Komponente der verfehlten Ziele
	`falseTrack`	Komponente der falschen Spuren

Nachfolgend sind die Anfangswerte der drei Parameter angegeben: `ClusteringThreshold` beträgt 5, `MinNumCellsPerCluster` 2 und `AssignmentThreshold` 30.Die Größe der Parameter lassen sich anpassen und die Verfolgungsergebnisse unter verschiedenen Parametereinstellungen beobachten.

Satz 1

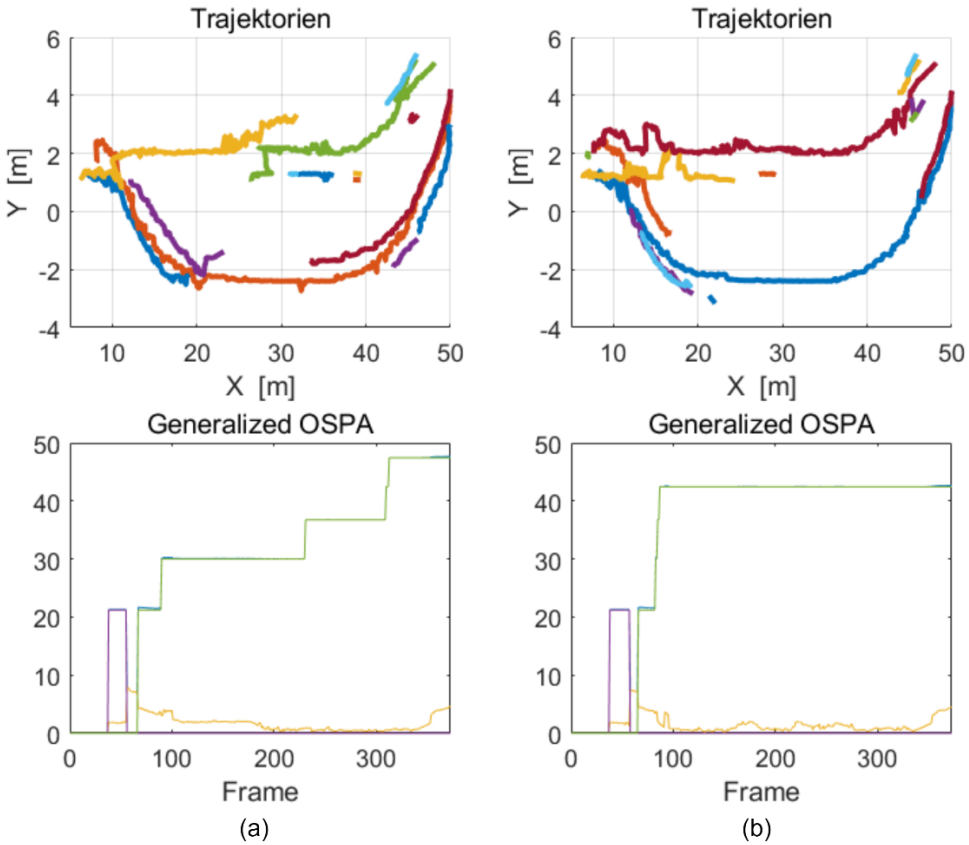


Abbildung 6.2 Diagramme über Trajektorien und GOSPA vom Satz 1

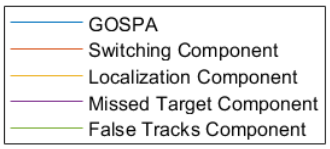


Abbildung 6.3 Legend von GOSPA

Tabelle 6.2 Eingestellte Parameter vom Satz 1

	Clustering Threshold	MinNumCells PerCluster	Assignment Threshold
a	5	2	4
b	5	2	3,4

In beiden Diagrammen sind viele Trajektorien zu beobachten. Diese Trajektorien zeigen insgesamt die Fahrbahnen der beiden Fahrzeuge, sollten jedoch zusammengeführt werden. Der Hauptgrund für die Vielzahl an Trajektorien liegt darin, dass die anfängliche minimale Anzahl der Cluster zu gering ist.

In Abbildung (a) steigt der Wert des Anteils der Fehlertrajektorien kontinuierlich in Stufen von 0 auf etwa 47, von Frame 66 bis Frame 316. In Abbildung (b) steigt der Wert des Anteils der Fehlertrajektorien von Frame 66 innerhalb von 25 Frames schnell von 0 auf etwa 47 und bleibt bis zum Ende der Verfolgung konstant.

Bei diesen beiden Parametern bleibt der Wert des Anteils der Fehlertrajektorien durchweg hoch. Zu Beginn, von Frame 37 bis Frame 58, treten Fälle von nicht erkannten Trajektorien auf, wobei der Wert des Fehlens etwa 21 beträgt. Danach erreicht der Wert des Lokalisierungsteils in den folgenden Frames den Höchstwert von etwa 7,6. Der Wert des Lokalisierungsteils bleibt relativ niedrig, mit einem Durchschnittswert von 2. Daraus lässt sich schließen, dass die Verfolgungsergebnisse bei diesen Parametern nicht ideal sind.

Satz 2

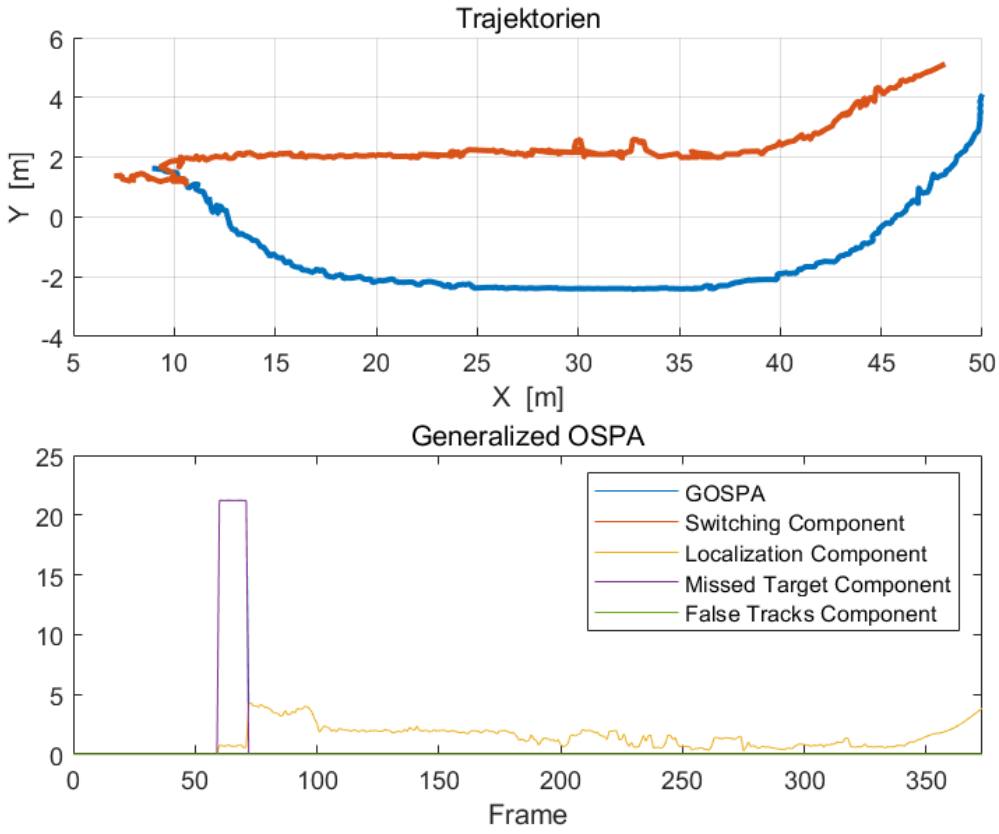


Abbildung 6.4 Diagramme über Trajektorien und GOSPA vom Satz 2

Tabelle 6.3 Eingestellte Parameter vom Satz 2

Clustering Threshold	MinNumCells PerCluster	Assignment Threshold
5	10	3,4

In der Abbildung 6.4 ist die Note der GOSPA insgesamt gut. Der Anteil der fehlerhaften Trajektorien bleibt über die gesamte Strecke hinweg bei 0, und der Wert des Lokalisierungsfehlers überschreitet nicht den Höchstwert von 4. Lediglich im Bereich von Frame 57 bis Frame 72 tritt ein Fehlermuster auf. Dies deutet darauf hin, dass diese Parametergruppe als Referenz für zukünftige Einstellungen verwendet werden kann.

Satz 3

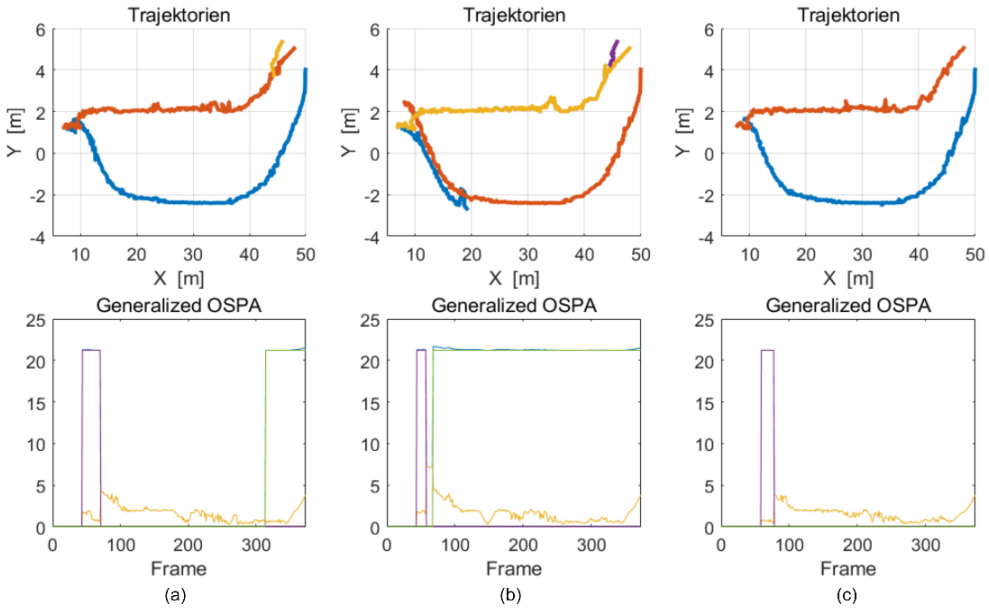


Abbildung 6.5 Diagramme über Trajektorien und GOSPA vom Satz 3

Tabelle 6.4 Eingestellte Parameter vom Satz 3

	Clustering Threshold	MinNumCells PerCluster	Assignment Threshold
a	2,5	5	3,4
b	2,5	5	4
c	2,5	10	4

In Abbildung 6.4(a) liegt der Wert der Fehlverfolgungen von Frame 43 bis Frame 70 bei etwa 21, was darauf hinweist, dass in diesem Zeitraum keine Trajektorien verfolgt wurden. Ab Frame 310 treten gelbe fehlerhafte Trajektorien auf.

In Abbildung 6.4(b) werden von Frame 43 bis Frame 58 keine Trajektorien erkannt. Anschließend steigt der Wert der Fehlverfolgungen ab Frame 66 auf etwa 21 und bleibt bis zum Ende der Verfolgung konstant. Zwischen Frame 58 und Frame 66 erreicht der Wert des Lokalisierungsfehlers den Höchstwert von 7. Die Hauptursache für diese fehlerhaften Trajektorien ist die zu geringe Anzahl

an Clustern. Kombiniert man Abbildung (a) und (b), lässt sich vernünftigerweise annehmen, dass bei denselben CT- und MCC-Werten die Reduzierung der AT zu einer Verbesserung der Verfolgungsergebnisse führen kann.

In Abbildung 6.4(c) ist der GOSPA-Score insgesamt gut. Der Wert der Fehlverfolgungen bleibt während der gesamten Strecke bei 0, und der Wert des Lokalisierungsfehlers überschreitet nicht den Höchstwert von 3. Fehlende Verfolgungen treten lediglich zwischen Frame 57 und Frame 78 auf. Zusammenfassend lässt sich feststellen, dass die Parametergruppe c eine vernünftige Einstellung darstellt.

6.2 Effizienz des Algorithmus

In diesem Abschnitt wird untersucht, wie verschiedene Faktoren die Recheneffizienz beeinflussen。Zunächst wird ein Vergleich der Recheneffizienz verschiedener Sensoren durchgeführt. Zur Bewertung des Einflusses der Berechnungsmethoden, der Gitterauflösung und zusätzlicher Funktionen auf die Laufzeiteffizienz werden Szenario 1, 3 und 6 von Livox als Beispiele herangezogen.

Vergleich der Laufzeit von Ouster/ Livox/ Leishen

Tabelle 6.5 Vergleich der Laufzeiteffizienz der drei Sensoren

Szenario	2	3	5	Sensor
FPS [Hz]	12,16	8,96	9,74	Ouster
	11,11	8,3	9,41	Livox
	8,26	6,51	7,58	Leishen

Es ist auffällig, dass die Laufeffizienz von Ouster und Livox ähnlich und deutlich höher als die von Leishen ist. Dies steht in engem Zusammenhang mit der Abtastrate der Sensoren. Die Abtastrate und die Berechnungseffizienz verhalten sich dabei umgekehrt proportional zueinander.

CPU/GPU

Tabelle 6.6 Vergleich der Laufzeiteffizienz mit CPU und GPU

Szenario	1		3		6	
Prozessor	mit CPU	mit GPU	mit CPU	mit GPU	mit CPU	mit GPU
FPS [Hz]	4,13	10,795	4,34	7,91	4,04	6,42

Bei der parallelen Implementierung wird eine Nvidia RTX 4050 verwendet. Es zeigt sich deutlich, dass die Nutzung einer GPU die Laufzeit des Algorithmus erheblich verbessern kann. Je weniger Trajektorien erkannt werden, desto höher ist die Rechengeschwindigkeit.

Resolution

Tabelle 6.7 Vergleich der Laufzeiteffizienz mit unterschiedlicher Resolution

Resolution	1	5	8	10	Szenario
FPS [Hz]	9,4	10,795	6,93	5,79	1
Verfolgungseffekt	--	+++	+++	+++	
FPS [Hz]	8,51	7,91	5,93	5,18	3
Verfolgungseffekt	---	++	++	-	
FPS [Hz]	7,18	6,42	5,7	4,59	6
Verfolgungseffekt	---	++	++	++	

In der Regel führt eine Auflösung von 1 zu einer leichten Verbesserung der Laufzeit. Diese niedrige Auflösung kann jedoch oft zu unbefriedigenden Verfolgungsergebnissen führen. Bei einer Auflösung von 8 sind die Verfolgungsergebnisse im Wesentlichen mit denen bei einer Auflösung von 5 vergleichbar, jedoch verringert sich die Laufgeschwindigkeit. Eine Auflösung von 10 ähnelt der von 8, zeigt jedoch schlechtere Verfolgungsergebnisse in Szenario 3. Insgesamt erweist sich eine Auflösung von 5 als die beste Wahl, da sowohl die Laufgeschwindigkeit als auch die Verfolgungsergebnisse optimal sind. Daher wird eine Auflösung von 5 gewählt.

mit zusätzlichen Funktionen

Tabelle 6.8 Vergleich der Laufzeiteffizienz mit zusätzlichen Funktionen

Funktion	Ansicht der Sensor	dynamische Karte	Analyse des Verschwindens	alle	Szenario
FPS [Hz]	5,37	1,597		1,25	1
	5,71	2,73		2,17	3
	4,62	1,85	1,9	0,2	6

Nach der Einführung der Aktualisierung der Sensormarkierungsinformationen alle 50 Frames wird ein merklicher Rückgang der Laufzeiteffizienz beobachtet. Die Effizienz in Szenario 1 sinkt von etwa 10 Hz auf die Hälfte, und in Szenario 3 fällt sie von 7,9 Hz auf 5,7 Hz. Im Vergleich dazu führt die Aktualisierung der dynamischen Karte bei jedem Frame zu einem erheblichen Rückgang der Laufzeiteffizienz. Der durch die Ausführung beider Funktionen verursachte Effizienzverlust ist im Wesentlichen vergleichbar mit dem durch die Aktualisierung der Punktwolken. Dies zeigt, dass ihrer Einfluss- nur einen kleinen Teil des gesamten Effizienzverlustes ausmacht. Bei Bedarf, wie etwa zu Beginn des Sensorbetriebs oder bei suboptimalen Verfolgungsergebnissen, wird ein zusätzliches Programm ausgeführt, um die Ursachen für die suboptimalen Ergebnisse zu identifizieren und die Parameter entsprechend zu optimieren.

6.3 Verfolgungsergebnisse

Vor der Analyse ist darauf hinzuweisen, dass in Szenario 01 die Punktwolkeninformationen des dritten Fahrzeugs, die von Leishen erfasst wurden, fehlerhaft sind. Daher wurde entschieden, diese Punktwolkeninformationen direkt herauszufiltern. Die Zeit in Sekunden ist gleich der Anzahl der Frames geteilt durch 10.

Die Parametereinstellungen unterscheiden sich beim Verfolgen von Fußgänger und Fahrzeugen. In der folgenden Tabelle 6.9 werden die Parameter aufgelistet, deren Wert durch Überlegungen bzw. Annahmen bestimmt sind. Wenn die Verfolgungsergebnisse weiter optimiert werden sollen, können der `ClusteringThreshold` und der `AssignmentThreshold` feinjustiert werden. Die Anpassungsbereiche sind wie folgt.

Tabelle 6.9 Übersicht der gewählten Parameterwerte

	`Clustering Threshold`	`MinNumCells PerCluster`	`Assignment Threshold`
Auto	3(2,5…4)	20	3,4(2,8…5)
Fußgänger	3	4	5,857

6.3.1 Verhalten der Punktwolke

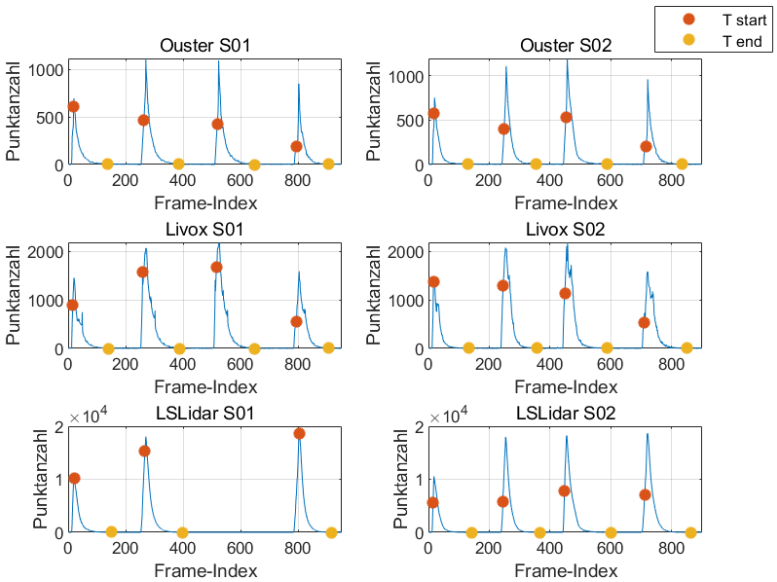


Abbildung 6.6 Diagramme über die Anzahl der von verschiedenen Sensoren erkannten Punkte

Abbildung 6.6 veranschaulicht das Verhalten der Punktwolke verschiedener Fahrzeuge in Szene 1 (links) und Szene 2 (rechts) in Bezug auf das Region of Interest (ROI). Die Darstellung ermöglicht eine detaillierte Analyse der Erstellung der Trajektorien. Die Trajektorien werden ausschließlich aus dynamischen Gitterzellen extrahiert, wodurch statische Objekte effizient herausgefiltert werden.

Dabei ist zu beachten, dass die Erstellung der Trajektorien nach dem Eintritt eines Fahrzeugs in das Gittergebiet mehrere Frames in Anspruch nehmen kann. Dies lässt sich auf zwei Hauptursachen zurückführen: Erstens gibt es eine Verzögerung bei der Klassifizierung einer Zelle als dynamisch. Zweitens erfordert der Prozess der Objektbestätigung mehrere Schritte, um die Trajektorie als zuverlässig zu etablieren.

Die spezifischen Werte der maximalen Punktanzahl sind zusammengefasst in Anhang 5.2 dargestellt. Leishen weist mit etwa 19.000 Punkten die größte maximale Punktanzahl auf, gefolgt von Livox mit maximal 2.200 Punkten und Ouster mit 1.200 Punkten. Die Abtastraten der Sensoren rangieren entsprechend von hoch nach niedrig: Leishen, Livox und Ouster. Eine höhere Abtastrate ermöglicht es, feinere Details des Objekts zu erfassen, insbesondere bei dynamischen Szenen. Dies führt jedoch gleichzeitig zu erhöhten Datenmengen und somit zu höheren Anforderungen an Speicher- und Verarbeitungsressourcen.

Deutlich wird auch, dass das erste Fahrzeug, der Twizy, die geringste Anzahl erfasster Punkte aufweist, was auf seine geringere Größe und somit die kleinere effektive Fläche im Erfassungsbereich des Sensors zurückzuführen ist. Eine überraschende Beobachtung wurde jedoch bei Livox und Ouster gemacht: Die Punktanzahl des vierten Fahrzeugs, des Porsche, ist signifikant geringer als die des zweiten Fahrzeugs, des Passat, und des dritten Fahrzeugs, des BMW i3, obwohl der Porsche das größte Volumen der vier Fahrzeuge besitzt. Im Folgenden werden mögliche Ursachen für dieses unerwartete Ergebnis diskutiert.

Die Reflexionseigenschaften der Fahrzeugoberfläche können die Qualität der vom Sensor erfassten Punktwolke erheblich beeinflussen. Unterschiedliche Farben weisen unterschiedliche Reflexions- und Absorptionseigenschaften gegenüber dem Laserlicht auf. Dunkle Objekte absorbieren in der Regel mehr Licht, während helle Objekte mehr Licht reflektieren, was die Erfassung beeinflussen kann. Zudem können glatte oder glänzende Oberflächen, wie Spiegel oder hochglänzende Lackierungen, zu Streueffekten und Reflexionen führen, die die Erkennungsgenauigkeit des Sensors verringern. Auch die Intensität und der Einfallswinkel des Umgebungslichts können die Lichtreflexion beeinflussen.

Darüber hinaus spielen die Position und der Winkel des Sensors eine entscheidende Rolle bei der Erfassung des Zielobjekts. Eine suboptimale Positionierung oder ein ungünstiger Winkel des Sensors kann dazu führen, dass nicht alle Details des Zielobjekts erfasst werden, was insbesondere bei größeren Objekten wie dem Porsche ausgeprägter sein kann.

Es ist festzustellen, dass die Auswirkungen der Veränderung der effektiven Erfassungsfläche

aufgrund der Bewegung des Zielobjekts auf die Datenerfassung bei allen Sensoren ähnlich sind. Kein Sensor zeigt bei bestimmten Fahrzeugformen eine signifikant überlegene Leistung bei der Erfassung der Punktwolkendaten. Typischerweise folgt die Punktanzahl zunächst einem raschen Anstieg, gefolgt von einem allmählicheren Rückgang nach Erreichen des Maximums. Im Endstadium, wenn die Punktanzahl gegen null sinkt, verläuft der Rückgang zunehmend flacher. Bei Livox zeigen sich jedoch während dieses Prozesses einige kleinere Unregelmäßigkeiten.

6.3.2 Belegungswahrscheinlichkeit der Gitterzellen

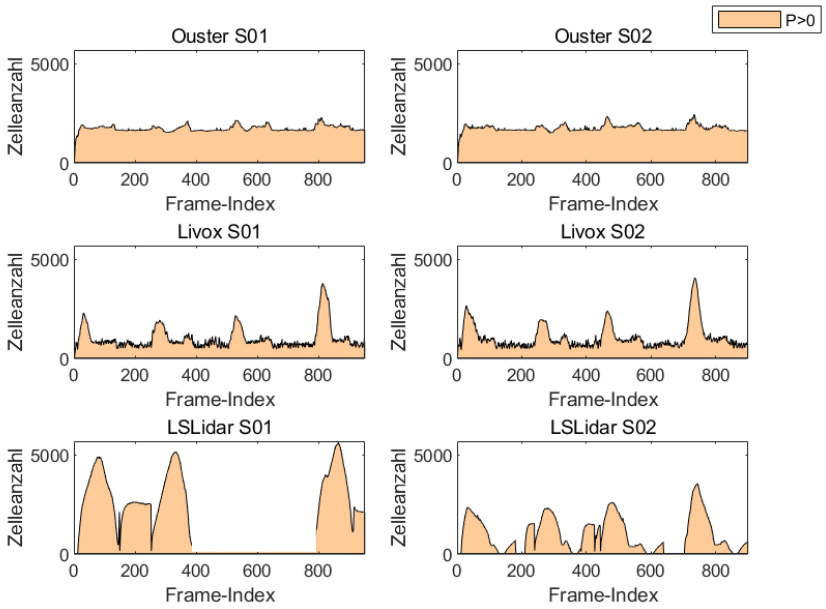


Abbildung 6.7 Diagramme über die Anzahl der Gitter mit P>0

Diese Diagramme in Abbildung 6.7 veranschaulichen die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0 für die Sensoren in den Szenarien 1 und 2. Es lässt sich beobachten, dass die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0 bei Ouster und Livox stets größer als null ist, selbst wenn keine Objekte im Interessensbereich erkannt werden. Diese Gitterzellen können als beobachtetes Rauschen interpretiert werden.

Bei Ouster bleibt die Anzahl der Gitterzellen in diesem Zeitraum konstant bei etwa 1500 bis 1600. Bei Livox liegt die Anzahl bei ungefähr 600. Im Vergleich dazu kann die Anzahl der Gitterzellen bei Leishen in diesem Zeitraum auf null sinken. Nach dem Auftreten eines Objekts in Szenario 2 steigt die Anzahl der Gitterzellen wieder von null auf 2300 bis 3500. In Szenario 1 erreicht die Anzahl der Gitterzellen einen Höchstwert von etwa 5500.

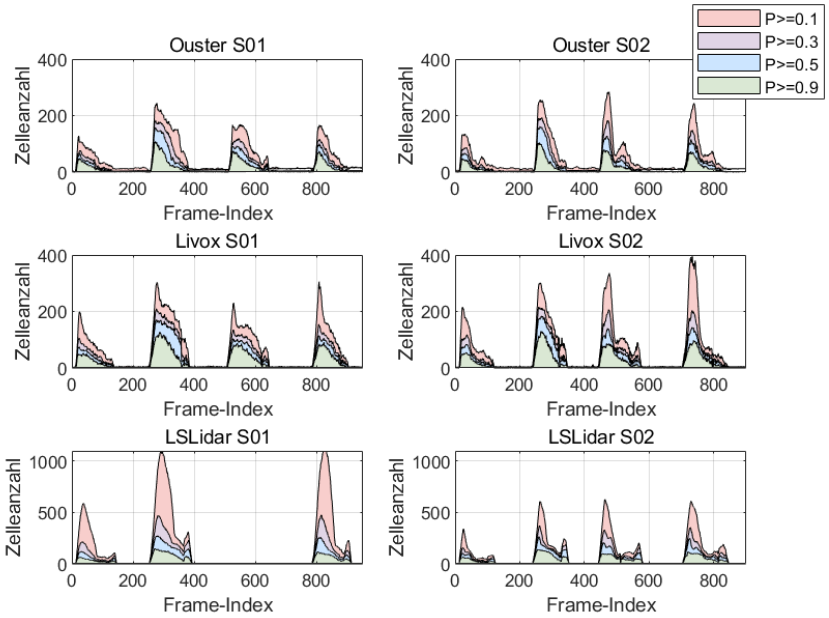


Abbildung 6.8 Diagramme über die Anzahl der Gitter mit den verschiedener Belegungswahrscheinlichkeiten

Abbildung 6.8 zeigt die Anzahl der Gitterzellen mit Belegungswahrscheinlichkeiten größer als 0,1, 0,3, 0,5 und 0,9 für die drei Sensoren in den Szenarien 1 und 2.

Bei Ouster bleibt die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0,9 konstant und stellt einen großen Anteil dar. Die Anzahl der Gitterzellen mit Belegungswahrscheinlichkeiten zwischen 0,1 und 0,3 ist ähnlich hoch wie die Anzahl der Gitterzellen mit Belegungswahrscheinlichkeiten größer als 0,9. Gitterzellen mit Wahrscheinlichkeiten zwischen 0,1 und 0,3 sowie zwischen 0,3 und 0,5 sind weniger vertreten. Im Szenario 1 von Livox zeigt sich ein ähnliches Muster wie bei Ouster.

Bei Leishen übertrifft die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit zwischen 0,1 und 0,3 die der anderen Sensoren. Danach folgt die Anzahl der Gitterzellen mit Belegungswahrscheinlichkeiten größer als 0,9. Die Anzahl der Gitterzellen mit Wahrscheinlichkeiten zwischen 0,3 und 0,5 sowie zwischen 0,1 und 0,3 ist vergleichbar.

Für die Verfolgung der Trajektorien sind Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0,9 erforderlich. Eine höhere Anzahl solcher Gitterzellen erhöht die Wahrscheinlichkeit für bessere Verfolgungsergebnisse.

6.3.3 Kalibrierung der Messdaten von Leishen

Nach der Betrachtung aller Verfolgungsergebnisse zeigt sich, dass die Trajektorien von Leishen im Vergleich zu den anderen beiden Sensoren erhebliche Abweichungen aufweisen. Nach sorgfältiger Analyse könnte die Ursache dafür eine leichte Neigung der Leishen-Sensoren sein, die zu Ungenauigkeiten führt. Um diese Annahme zu überprüfen, wurde auch eine globale Ansicht erstellt. Im Folgenden wird diese Abweichung anhand von Szenario 1 näher erläutert.

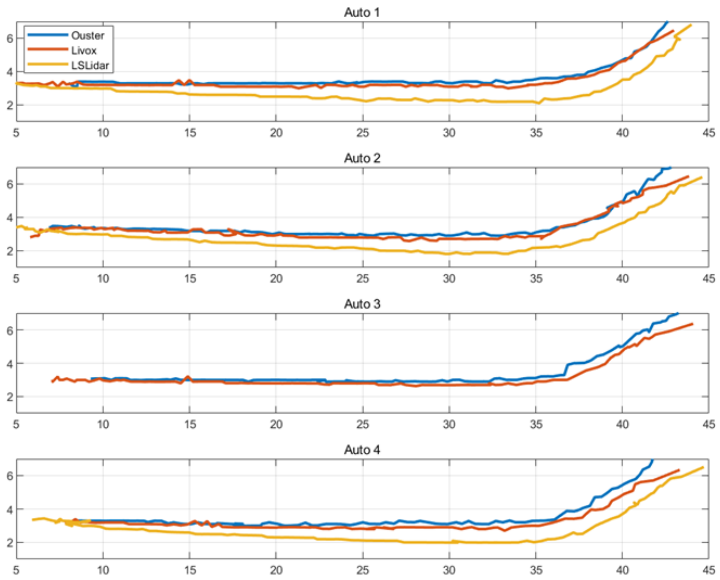


Abbildung 6.9 Unkorrigierte Verfolgungsergebnisse verschiedener Sensoren in Szenario 1

Durch die Beobachtung der Abbildung 6.9 ist deutlich zu erkennen, dass die Trajektorien von Ouster und Livox nahezu übereinstimmen, wobei nur geringfügige Abweichungen vorhanden sind. Solche Abweichungen können als akzeptabel angesehen werden und sind wahrscheinlich auf Unterschiede in den erfassten Punktwolkendaten zurückzuführen.

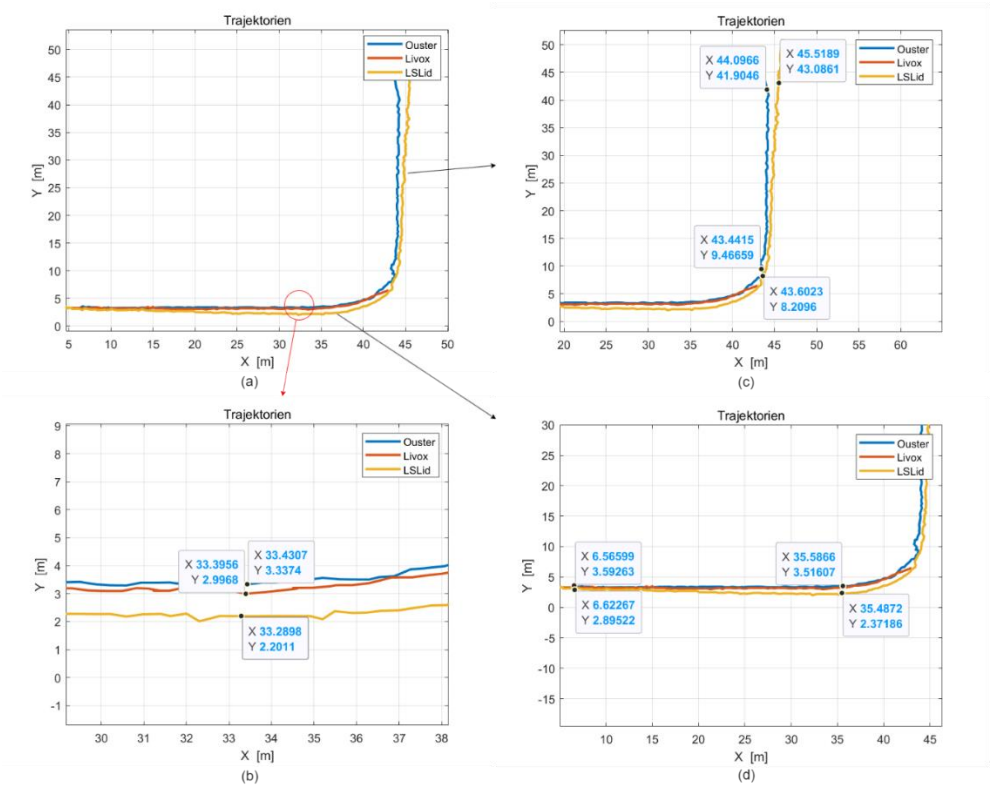


Abbildung 6.10 Darstellung der Abweichung der Trajektorie vom Leishen

Abbildung 6.10(a) zeigt die Panoramaansicht der Trajektorie des ersten Fahrzeugs im ersten Szenario. Abbildung (b) zeigt eine lokale Vergrößerung vor der Kurve des Fahrzeugs. Es ist deutlich zu erkennen, dass in der Nähe von x = 33 m die Abweichung in y-Richtung zwischen den Ergebnissen von Ouster und Livox etwa 30 cm beträgt, während die Abweichung zu Leishen 1,1 m beträgt. Abbildung (d) zeigt die Trajektorie des Fahrzeugs vor der Kurve. Es ist klar ersichtlich, dass die y-Achsen-Abweichung bei Ouster beim Fahren von x = 6,6 m bis x = 35,5 m 0,08 m nicht überschreitet. Im Vergleich dazu beträgt die Abweichung der von Leishen erkannten Trajektorie etwa 0,52 m, was ungefähr dem Siebenfachen der Abweichung von Ouster entspricht.

Abbildung 6.10(c) zeigt die Trajektorie des Fahrzeugs nach der Kurve. In den von Leishen erkannten Ergebnissen hat sich das Fahrzeug nach einer gewissen Zeit des Fahrens nach der Kurve relativ zu seinem Wendepunkt entlang der x-Achse um 1,92 m verschoben, was ungefähr der Verschiebung von einem halben Fahrstreifen entspricht. Dies steht im Widerspruch zu der Tatsache, dass das Fahrzeug im Experiment stets entlang der Mittellinie des linken Fahrstreifens gefahren ist.

Daraus lässt sich die Schlussfolgerung ableiten, dass eine Abweichungskorrektur erforderlich ist. Die Verfolgungsergebnisse von Leishen müssen um 1,64 Grad entlang der x-Achse rotiert werden, um die Korrektur vorzunehmen. Die Ergebnisse der Kalibrierung sind wie folgt:

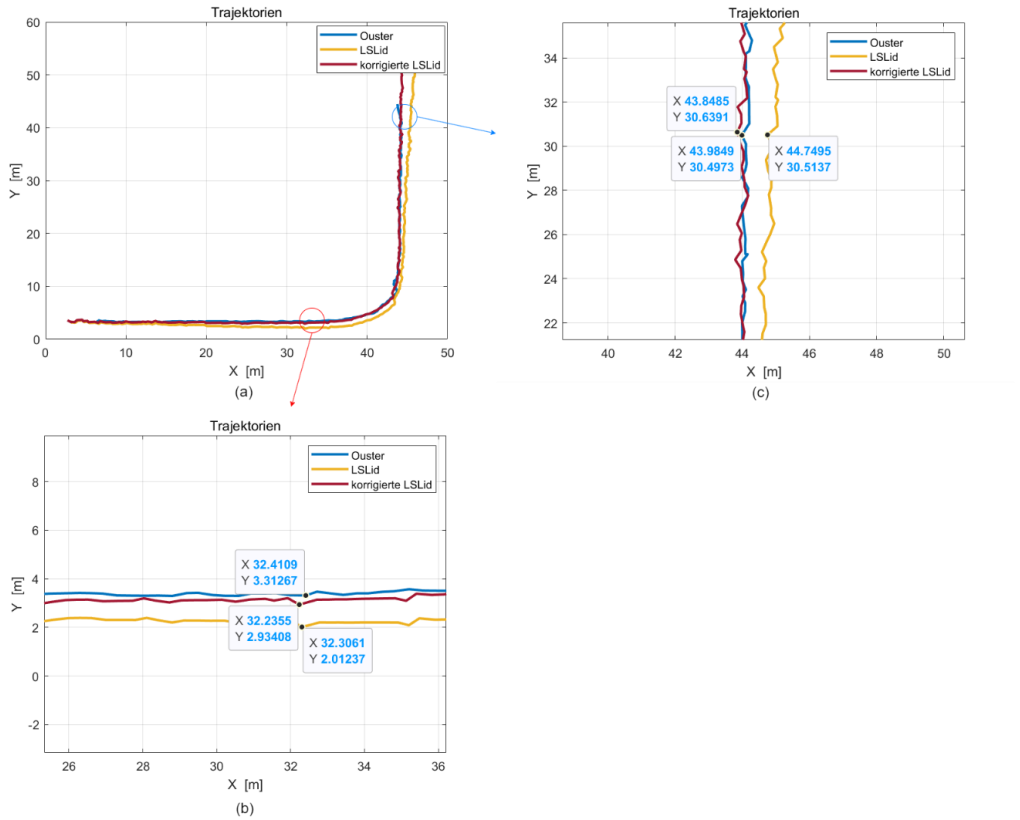


Abbildung 6.11 Darstellung der korrigierten Trajektorien vom Leishen

Aus den Abbildungen 6.11(b) und 6.11(c) lässt sich schließen, dass nach der Anpassung die Abweichung der Trajektorie von Leishen von der Trajektorie von Ouster 0,4 m nicht überschreitet. Die Ablenkung der Leishen-Trajektorie könnte durch einen Installationsfehler des Sensors verursacht worden sein. Das bedeutet, dass die Vorderseite des Leishen-Sensors um etwa 1,6 Grad zur negativen y-Achse im Vergleich zur x-Achse geneigt war.

6.3.4 Trajektorien des Einzelziels

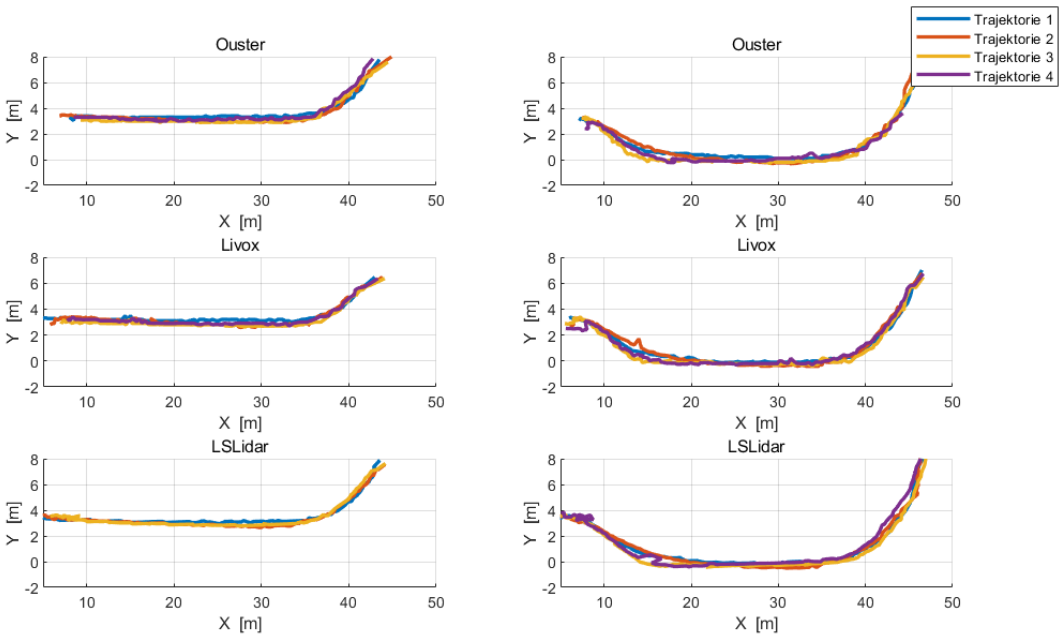


Abbildung 6.12 Verfolgungsergebnisse verschiedener Sensoren in Szenario 1 und 2

Auf der linken Seite des Bildes 6.12 sind die Verfolgungsergebnisse aus Szenario 1 und auf der rechten Seite die Ergebnisse aus Szenario 2 dargestellt. Alle drei Sensoren ermöglichen eine ausgezeichnete Zielverfolgung, und die Trajektorien der vier verfolgten Fahrzeuge stimmen weitgehend überein. Bei Verwendung der Livox- und Leishen-Sensordaten kann der Tracker die Trajektorien früher erkennen, wobei der Beginn der Trajektorien etwa 1 Meter weiter in Richtung der x-Achse liegt als bei Ouster. Im Vergleich zu Livox zeigen die von Ouster und Leishen verfolgten Trajektorien einen geschmeidigeren Kurvenverlauf während der Fahrzeugbewegungen in den Kurven. Bei Livox ähnelt diese Phase eher einer geraden Linie mit konstantem Neigungswinkel.

Im Vergleich zu den Verfolgungsergebnissen aus Szenario 1 zeigt Szenario 2 einige leichte Biegungen in den Trajektorien. Diese Schwankungen treten nicht nur zu Beginn der Zielverfolgung auf, sondern auch bei Trajektorie 2 in Livox etwa bei x = 13 m und bei Trajektorie 4 in Leishen etwa bei x = 15 m.

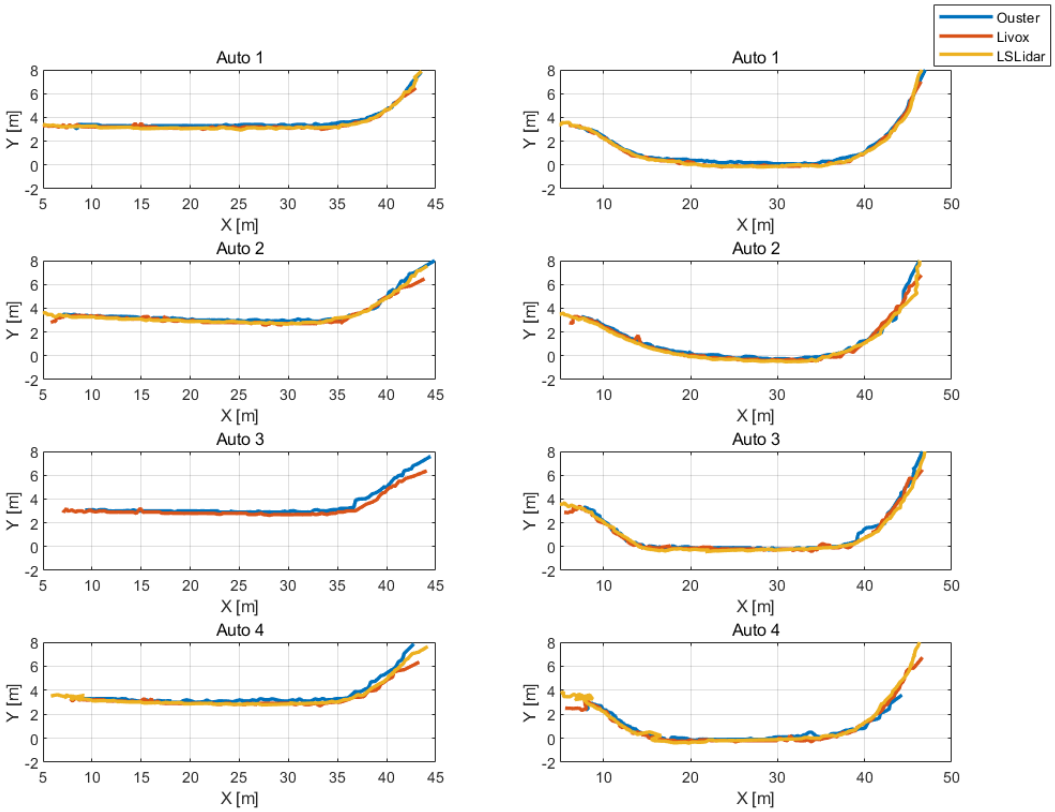


Abbildung 6.13 Verfolgungsergebnisse verschiedener Fahrzeuge in Szenario 1 und 2

In sowohl Szenario 1 als auch Szenario 2 stimmen die von den drei Sensoren bereitgestellten Fahrzeugtrajektorien weitgehend überein. In Szenario 1 sind die Trajektorien der drei Sensoren vor dem Abbiegen jedes Fahrzeugs vollständig deckungsgleich. Bei Livox zeigt sich jedoch am Ende der Verfolgung, dass die Trajektorien der Fahrzeuge etwa 2 Meter nach unten in y-Richtung verschoben sind im Vergleich zu Ouster.

In Szenario 1 weist Fahrzeug 2 im Vergleich zu den anderen Fahrzeugen während der Geraden eine leichte Neigung in Richtung der negativen y-Achse auf. In Szenario 2 ist deutlich zu sehen, dass die Fahrzeuge 1, 3 und 4 bei etwa x = 10 m eine starke Kurve machen und bei x = 15 m (y = 0 m) wieder geradeaus fahren. Fahrzeug 2 vollzieht eine sanftere Kurve und beendet diese bei etwa x = 20 m. An diesen Trajektorien lässt sich erkennen, dass die Fahrgewohnheiten der Fahrer leicht variieren.

In Szenario 2 stimmen die Trajektorien von Fahrzeug 1 vollständig überein. Bei Fahrzeug 2 gibt es geringfügige Abweichungen zwischen den drei Sensoren. Fahrzeug 3 zeigt während der Abbiegephase eine deutliche Biegung in den von Ouster erkannten Trajektorien. Die Verfolgung von Fahrzeug 4 weist ebenfalls leichte Abweichungen zu Beginn und am Ende der Trajektorie auf.

Die Verfolgungsanalyse von Szenario 2 kann auch unter Einbeziehung des Gierwinkels durchgeführt werden. Es zeigt sich, dass die von den drei Sensoren erkannten Gierwinkel gut übereinstimmen.

Dies deutet darauf hin, dass alle drei Sensoren eine hohe Genauigkeit und Präzision bei der Erkennung des Gierwinkels aufweisen. Die Geschwindigkeiten der vier Fahrzeuge sind im Anhang 5.3 zusammengefasst.

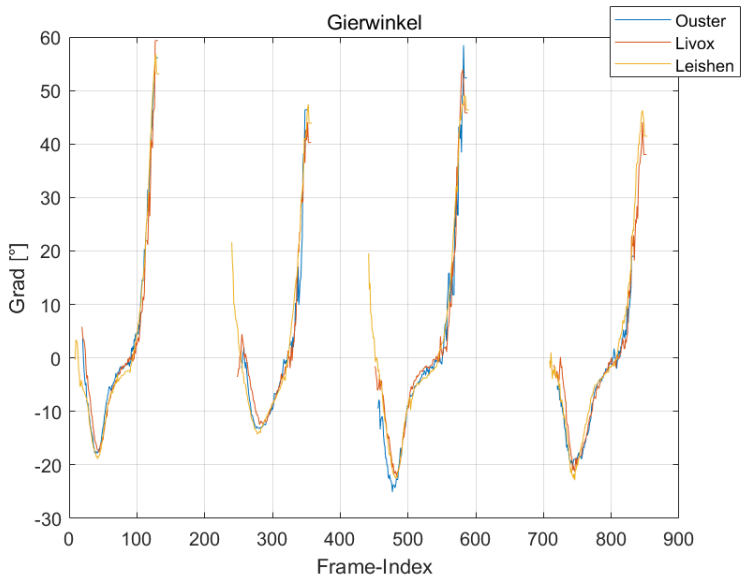


Abbildung 6.14 Diagramm über die Veränderung des Gierwinkels in Szenario 2

Die Verfolgungsergebnisse eines einzelnen Fußgängers befinden sich in Anhang 5.4 und 5.6.

6.3.5 Trajektorien mehrerer Ziele

Szenario 3

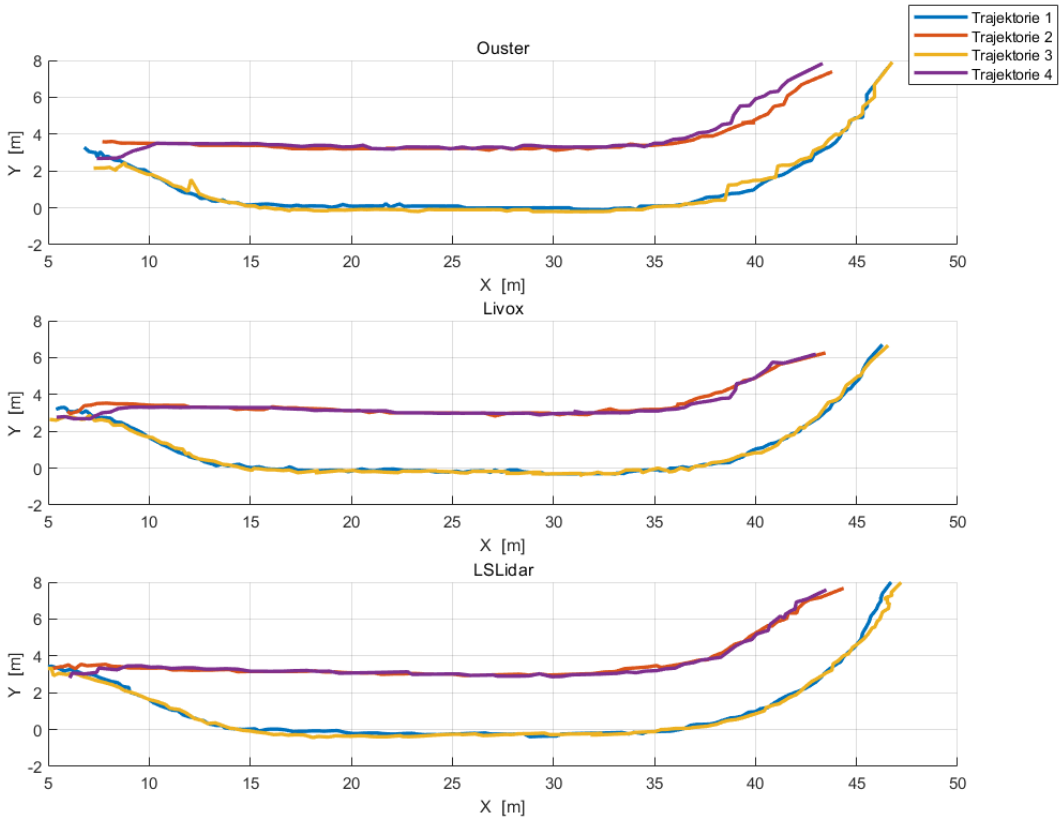


Abbildung 6.15 Verfolgungsergebnisse verschiedener Sensoren in Szenario 03

Insgesamt zeigen die erkannten Trajektorien eine hohe Übereinstimmung. Allerdings weicht die Trajektorie von Fahrzeug 4 bei Ouster und Livox im Vergleich zu Fahrzeug 2 ab. Am Ende der Verfolgung weist die Trajektorie von Fahrzeug 4 bei Ouster und Livox deutliche Unregelmäßigkeiten auf.

Bei allen drei Sensoren zeigt die Trajektorie von Fahrzeug 4 in der Einfahrphase eine Verschiebung von etwa y = 3 m auf y = 3,5 m nach einer kurzen Strecke. Tatsächlich fuhr Fahrzeug 4 jedoch geradeaus und führte keine derartigen Bewegungen aus. Dieses Verhalten ist auf die große Größe des Fahrzeugs, ein Porsche-Modell, zurückzuführen, das von Anfang an nicht effektiv zentriert werden konnte.

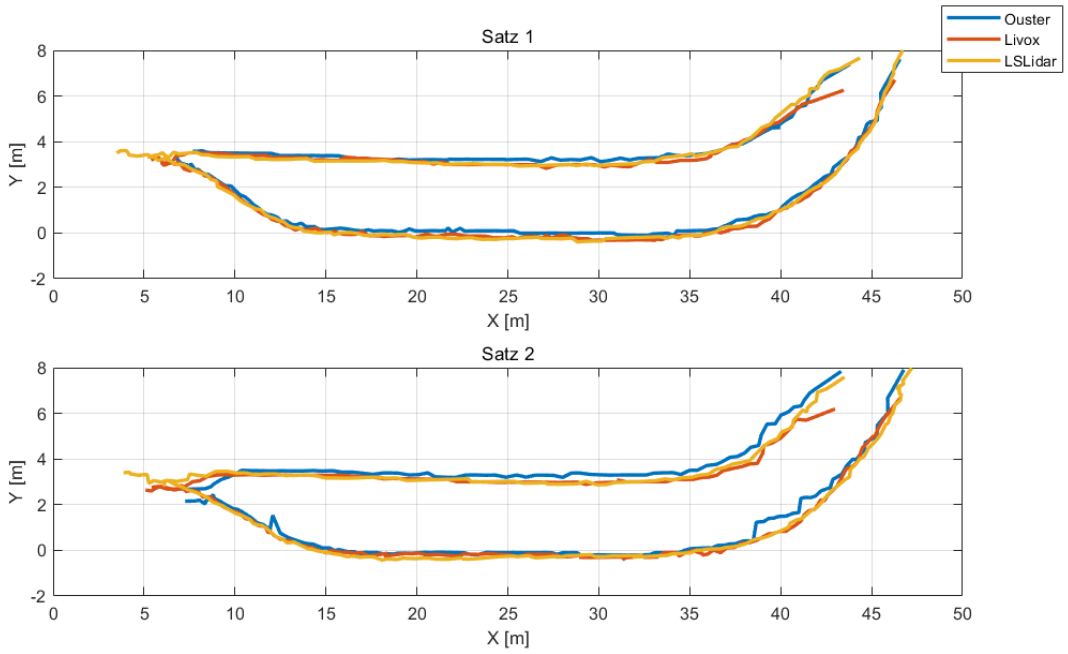


Abbildung 6.16 Verfolgungsergebnisse verschiedener Sätze der Fahrzeuge in Szenario 03

Die Verfolgungsergebnisse der ersten Gruppe, bereitgestellt von Ouster und Leishen, zeigen eine hohe Übereinstimmung. In den Ergebnissen der ersten und zweiten Gruppe von Livox weist die obere Fahrzeugtrajektorie am Ende stets eine nach unten verschobener Distanz im Vergleich zu den anderen beiden Sensoren auf. Die Verfolgungsergebnisse der zweiten Gruppe von Livox und Leishen sind weitgehend konsistent. Die von Ouster bereitgestellten Ergebnisse stimmen im ersten Abschnitt größtenteils überein, zeigen jedoch ab x = 38 m größere Schwankungen im zweiten Abschnitt.

Szenario 6

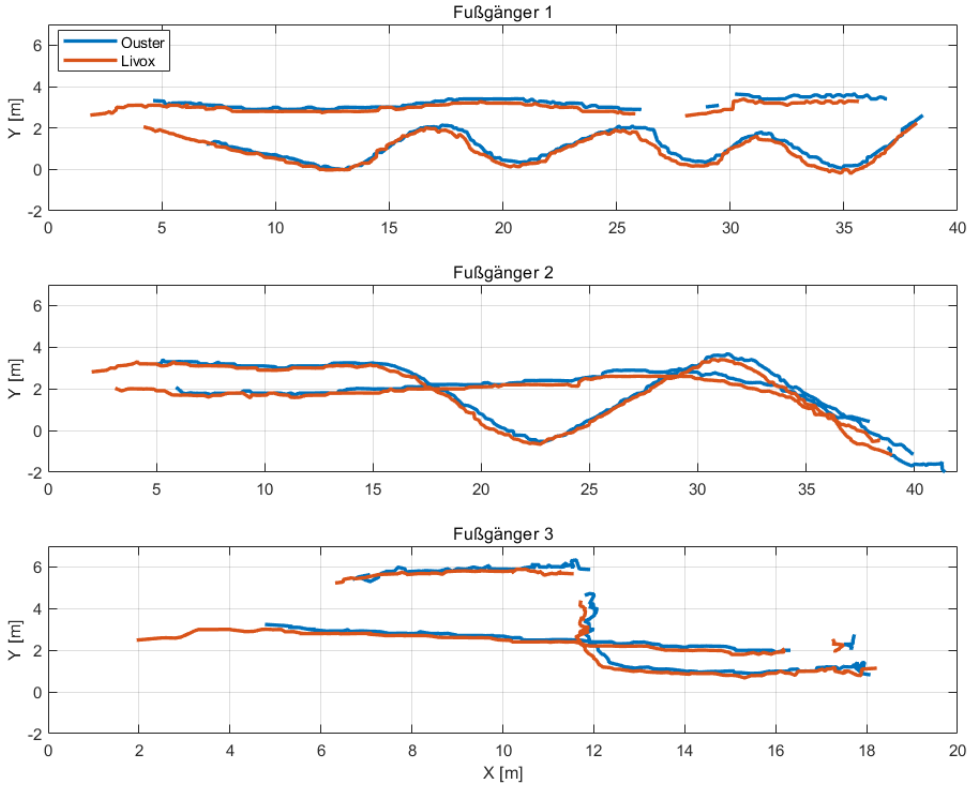


Abbildung 6.17 Verfolgungsergebnisse verschiedener Fußgänger in Szenario 06

Analyse

Abbildung 6.17 liefert Informationen über die Verfolgungsergebnisse verschiedener Fußgänger von Ouster und Livox in Szenario 06. Die Trajektorien aller drei Fußgänger sind nicht durchgängig. Die Trajektorie von Fußgänger 1 ist bei x = 28 m und x = 38,3 m unterbrochen. Die Trajektorie von Fußgänger 2 weist bei x = 39 m eine Unterbrechung auf. Die Trajektorie von Fußgänger 3 ist bei x = 11,4 m und x = 18,2 m unterbrochen. Die detaillierten Ergebnisse eines Erkennungsalgorithmus sind in Anhang 5.6 zu finden.

Im Folgenden werden die Gründe für die Unterbrechungen der Trajektorien näher erläutert. Die Ursachen für diese Unterbrechungen lassen sich in drei Punkte unterteilen:

- Beim Erfassen der Punktwolkendaten durch den Sensor kann ein nahe am Sensor befindliches Ziel die anderen beiden vollständig verdecken. Dies führt dazu, dass die Punktwolkendaten der verdeckten Objekte vollständig fehlen. Es ist jedoch wichtig zu betonen, dass ein Ziel, solange es nicht vollständig verdeckt ist und noch teilweise Punktwolkendaten erfasst werden können, nicht übersehen wird und weiterhin angezeigt bleibt.

- Da der Bewegungszustand der Objekte auf der Grundlage einer 2D-Gitterkarte geschätzt wird, erfordert die Aktualisierung der Trajektorien, dass die Zellen dynamisch bleiben. Selbst kurze Pausen der Ziele oder Geschwindigkeiten nahe 0 können dazu führen, dass sie verschwinden, wodurch ihre Trajektorien unterbrochen werden.
- Verlässt das Ziel den Interessenbereich und kehrt später zurück, wird seine Trajektorie als eine neue Trajektorie gekennzeichnet.

Die unterbrochene Trajektorie vom Fußgänger 1 wurde durch die Verdeckung seiner Punktwolken-informationen verursacht. Es gibt zwei Möglichkeiten für das wiederauftauchende Objekt in der Nähe des Bereichs, in dem Fußgänger 2 verschwand: Entweder hat sich er aus dem ROI-Bereich herausbewegt und ist später wieder aufgetaucht, oder ein neues Objekt ist in den ROI-Bereich in der Nähe des Verschwindens von Fußgänger 2 eingetreten. Wenn die Anzahl der zu verfolgende Ziele unbekannt ist und es keine weiteren höheren Informationen gibt, ist es schwierig, das Objekt nur anhand der Belegungswahrscheinlichkeit und der Position der Zellen zu identifizieren. Bestimmte visuelle Merkmale können verwendet werden, um festzustellen, ob es sich beim neuen Objekt um dasselbe handelt wie das verschwundene. In dieser Arbeit wird dies jedoch nicht weiter untersucht. In dem Experiment, bei dem die Anzahl der verfolgten Objekte bekannt ist und auf drei Personen festgelegt wurde, kann diese Information genutzt werden, um die Trajektorien durch Ausschluss und den Vergleich der Positionen des Zielverschwindens mit den neuen Zielen zu verwalten. Dies führt dazu, dass die Trajektorie 6 als zum Fußgänger 2 gehörend identifiziert wird. Die unterbrochene Trajektorie von Fußgänger 3 wurde durch mehrmals Pause während seiner Bewegung verursacht.

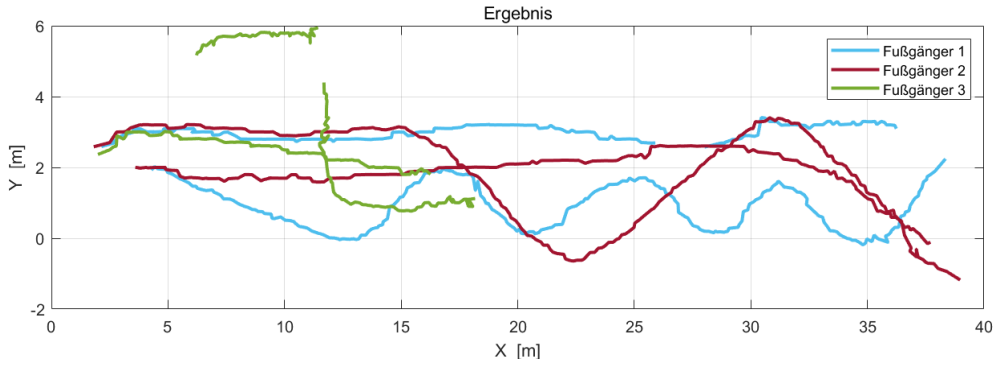


Abbildung 6.18 Trajektorien von Livox in Szenario 06

Auftreten und Umgang mit dem Zielverlust

Wenn sich Objekt 1 und Objekt 2 sehr nah beieinander befinden, kann es aufgrund des Blickwinkels des Sensors vorkommen, dass keine Blockierung der Punktwolke auftritt. Wenn jedoch Objekt 1

näher am Sensor ist und sich Objekt 2 in dessen totem Winkel befindet, wird Objekt 2 zwangsläufig verdeckt. Die Verdeckung kann anhand der Belegungswahrscheinlichkeit abgeschätzt werden. Wenn in der Nähe von Zellen mit einer Belegungswahrscheinlichkeit von etwa 0,5 keine Zellen mit einer hohen Belegungswahrscheinlichkeit vorhanden sind, bedeutet dies, dass kein Objekt oder dessen Trajektorie korrekt erkannt wurde. Diese unklare Situation kann auftreten, wenn der Bereich nicht erkannt wurde. Eine weitere Möglichkeit besteht darin, dass das Zielobjekt verdeckt ist und daher nicht erkannt werden kann. Basierend auf dem Zeitpunkt der teilweisen Verdeckung und der geschätzten Geschwindigkeit kann der Moment einer möglichen Kollision ungefähr vorhergesagt werden.

Im Folgenden wird das grundlegende Verfahren erläutert, während der konkrete Implementierungsprozess in Abbildung 6.19 veranschaulicht wird. Wenn eine zufällige Anzahl von Objekten in einem Interessensbereich erscheint, erstellt der Tracker Trajektorien und generiert entsprechende Objekt-IDs. Diese Objekte werden von "unbekannt" auf "bekannt" gesetzt. Wenn ein bekanntes Objekt bereits eine Trajektorie besitzt, diese jedoch nach einem bestimmten Frame nicht mehr verfolgt wird, kann dies folgende Gründe haben:

1. **Das Objekt bewegt sich aus dem ROI-Bereich (Erkennungsbereich) heraus.**
   Diese Situation lässt sich anhand der Position des Objekts im letzten Frame, in dem seine Trajektorie erkannt wurde, feststellen. Befindet sich das Objekt am Rand des Interessensgebiets, handelt es sich um einen normalen Zielverlust.

2. **Das Punktwolken-Informationssignal des Objekts wurde vollständig verdeckt**, sodass es nicht mehr erkennbar ist. Es ist bekannt, dass selbst bei teilweise erkannten Punktwolken die Trajektorie des Objekts immer noch verfolgt werden kann. Die vollständige Verdeckung ist ein schrittweiser Prozess, der von einer teilweisen zu einer vollständigen Verdeckung übergeht. Wenn die Auflösung des Sensors normal ist, kann dieser Prozess aufgezeichnet werden. Selbst bei plötzlichem Verlust der Punktwolkeninformationen, von vollständiger Sichtbarkeit in einem Frame zu vollständiger Verdeckung im nächsten, liefert der Frame vor dem Zielverlust noch viele nützliche Informationen. Durch den Vergleich der Objektdaten zwischen dem letzten und dem aktuellen Frame kann festgestellt werden, welches Zielobjekt vollständig verdeckt wurde.

3. **Zielstopp verursacht die Unterbrechung der Trajektorien**
   In dieser Situation kann durch den Vergleich der Belegungszellen des letzten Frames und des aktuellen Frames, in dem das Ziel verschwunden ist, überprüft werden, ob es sich um einen tatsächlichen Stopp des Objekts handelt. In solchen Fällen kann die letzte bekannte Trajektorie

weiterhin verfolgt werden, indem die hochbelegten Zellengruppen weiter überwacht und markiert werden, bis das Objekt wieder erkannt wird. Wenn das Ziel in der Nähe der zuvor belegten Zellen erkannt wird und sich innerhalb einer definierten Schrittweite bewegt, kann angenommen werden, dass es sich um dasselbe Objekt handelt. Die Objekt-ID kann entweder angepasst oder im Analyseabschnitt zur weiteren Bearbeitung notiert werden.

4. **Fehlerhafte Erkennung.**

   Es ist unwahrscheinlich, dass eine erkannte Trajektorie aufgrund von Parameterkonfigurationen nicht mehr erfasst wird, obwohl die Punktwolkeninformationen des Objekts noch vorhanden sind.

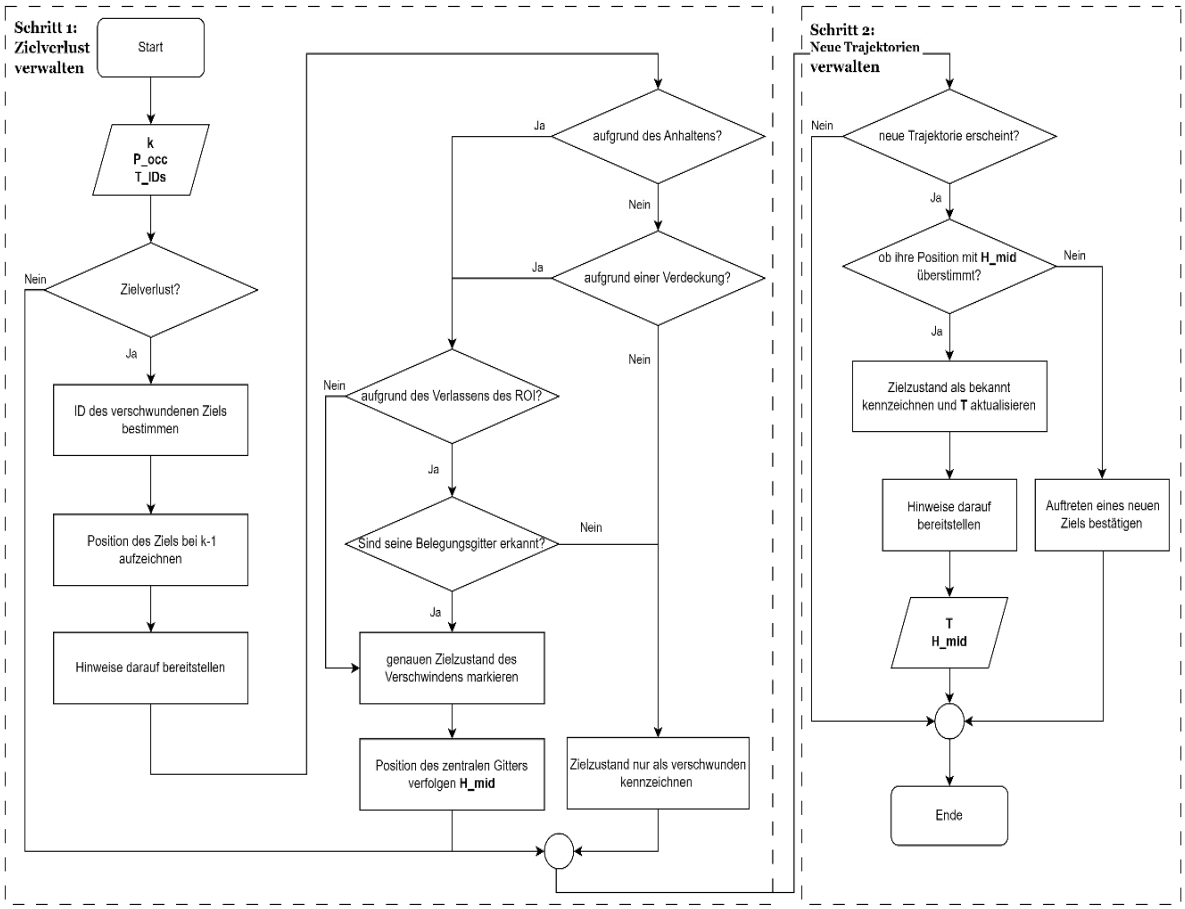


Abbildung 6.19 Programmablaufplan der Erkennung des Zielverlusts

Tabelle 6.10 Variable über die Erkennung des Zielverlusts

Variable	Beschreibung
P_occ	Stellt der Belegungswahrscheinlichkeit jeder Gitterzelle dar.
T	Erfasst die Trajektorien, die zu jedem Ziel gehört.
H_mid	Erfasst den Zeilenindex und den Spaltenindex vom zentralen belegten Gitterzelle.

7 Fazit und Ausblick

Dieser Arbeit kombiniert Bayes-Schätzung und FISST, um die grundlegenden Inhalte der Partikelfiltertheorie umfassend darzustellen. Auf Basis der Theorie der RFS und der Partikelfilter wurde ein Zielverfolgungsalgorithmus für verschiedene Szenarien entwickelt, der von der Zielerkennung über die Zustandsabschätzung bis hin zum Management der Trajektorien reicht. Gleichzeitig wurden grundlegende Informationen extrahiert und mit den Entscheidungen auf höherer Ebene kombiniert, um die Zielverschwinden-Situation zu bewerten.

Der entwickelte Algorithmus kann die Unsicherheit und dynamischen Veränderungen der Zielanzahl auf effektive Weise handhaben. Er verfügt über eine hohe Robustheit, insbesondere in komplexen Szenarien und bei Verdeckung. Zudem zeigt er eine hohe Anpassungsfähigkeit an verschiedene Szenarien.

Aber der Nachteil dieses Algorithmus ist nicht zu übersehen, wie in Abschnitt 5.1 erwähnt. Da nicht alle Teile des Fahrzeugs von dem LiDAR erfasst werden, ändert sich die erfasste Punktwolke des Ziels ständig mit der Bewegung des Objekts. Dies führt dazu, dass der Tracker die Form des Zielobjekts fortlaufend neu schätzt. Daher ist es notwendig, die Form des Objekts entweder im Voraus oder zu einem bestimmten Zeitpunkt festzulegen, um die genannten Störungen zu vermeiden.

In Abschnitt 6.1 wurde bereits erwähnt, dass in dieser Arbeit simulierte Experimente zur Bewertung der Wirksamkeit des Algorithmus herangezogen werden, da es an echten Werten mangelt. Um direkte Experimente mit realen Szenarien durchzuführen, müsste ein CAN-BUS-System integriert werden. Dabei wird im Experiment zunächst die Startposition jedes Fahrzeugs festgelegt, und durch die Erfassung von Geschwindigkeit und Gierwinkel während des Versuchs können die echten Werte ermittelt werden. Alternativ könnte GPS-Daten direkt verwendet werden. In zukünftigen Experimenten könnte die synchrone Erfassung von Positions-, Geschwindigkeits- und Gierwinkeldaten implementiert werden, um eine quantifizierte Bewertung der Verfolgungs-ergebnisse zu ermöglichen.

Zukünftige Arbeiten könnten darauf abzielen, die Leistung des Managements der Trajektorien weiter zu verbessern, zum Beispiel durch die Beschaffung zusätzlicher Informationen über die verfolgten Ziele (wie Doppler, Farbe, Textur). In diesem Fall wird die Unterscheidbarkeit der Zieltrajektorien erhöht. Da die Verwaltung der Trajektorien ein Problem der gemeinsamen Schätzung und Entscheidung darstellt, ist es mit vielen Unsicherheiten bei den Entscheidungen konfrontiert. Deshalb sind zusätzliche Informationen oder spezielle Entscheidungsregeln

erforderlich.

Zukünftig könnte auch die Integration von Informationen aus mehreren Sensoren und sogar unterschiedlichen Filtern in Betracht gezogen werden. Bei der Informationsfusion ist es wichtig, verschiedene Informationen zeitgerecht, angemessen und sinnvoll zu nutzen, um die Verfolgungsleistung zu verbessern. Es sollte angestrebt werden, Sensoren mit unterschiedlichen Schwerpunkten zu kombinieren, wie die erwähnten Doppler-, Farb- und Textursensoren. Bei möglichst geringer Informationsredundanz sollen ihre Vorteile kombiniert und eine Synergieeffekt erzielt werden.

Literatur- und Quellenverzeichnis

[1] D. Nuß:

A random finite set approach for dynamic occupancy grid maps. Ulm: Universität Ulm, 2016. ISBN 978-3-941543-28-7

https://oparu.uni-ulm.de/server/api/core/bitstreams/317ec699-b984-4c63-9e33-91862bea49ff/content

[2] T. Li, H. Fan und S. Sun:

Particle filtering: Theory, approach, and application for multitarget tracking. Acta Automatica Sinica, 2015, 41. Jg., Nr. 12, S. 1981-2002.
https://www.researchgate.net/Particle_filtering_Theory_approach_and_application_for_multitarget_tracking/links/59f464a90f7e9b553eba9ba6/Particle-filtering-Theory-approach-and-application-for-multitarget-tracking.pdf

[3] B. Shan und X. Yang:

The Research Progress on Multi-Extended Target Tracking Based on Random Finite Sets. Control and Decision, 2017, 32. Jg., Nr. 6, S. 961-966.

http://kzyjc.alljournals.cn/kzyjc/article/abstract/20170601

[4] R. Mahler:

Random sets: Unification and computation for information fusion-a retrospective assessment. In: Proceedings of the Seventh International Conference on Information Fusion. I, 2004. S. 1-20.

https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=eb276535cf1f422828d5e8050883a2deac6cfeb4

[5] R. Mahler:

Nonadditive probability, finite-set statistics, and information fusion. In: Proceedings of 1995 34th IEEE Conference on Decision and Control. IEEE, 1995. S. 1947-1952.

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=480631

[6] K. Sentz und S. Ferson:

Combination of evidence in Dempster-Shafer theory. PDF (online).2002.

https://www.osti.gov/servlets/purl/800792

[7] o.V.:

LIVOX Wiki. 2. Introduction to LIVOX Scanning Patterns. Webseite. 2020.

https://livox-wiki-en.readthedocs.io/en/latest/introduction/livox_scanning_pattern.html

(besucht am 24. 09. 2024).

[8] o.V.:

Lixov Horizion User Manual. PDF (online). o.J.

https://www.livoxtech.com/3296f540ecf5458a8829e01cf429798e/assets/horizon/Livox%20Horizon%20user%20manual%20v1.0.pdf

(besucht am 24. 09. 2024).

[9] o.V.:

Ouster OS1 Datasheet. PDF (online). o.J.

https://data.ouster.io/downloads/datasheets.pdf

[10]o.V.:

Leishen 128S2 Introduction. PDF (online). 2022.

https://www.lslidar.com/wp-content/uploads/2022/07/LSS2-Product-Introduction.pdf

[11]o.V.:

Lidar Toolbox: Design, analyze, and test lidar processing systems. Webseite. o.J.
https://de.mathworks.com/help/lidar/lidar_toolbox

[12]o.V.:

Sensor Fusion and Tracking Toolbox: Design, simulate, and test multisensor tracking and positioning systems. Webseite. o.J.

https://de.mathworks.com/help/fusion/Sensor_Fusion_Toolbox

[13]o.V.:

Parallel Computing Toolbox: Perform parallel computations on multicore computers, GPUs, and computer clusters. Webseite. o.J.

https://de.mathworks.com/help/parallel-computing/parallel_Computing_Toolbox

[14]o.V.:

Automated Driving Toolbox: Design, simulate, and test ADAS and autonomous driving systems. Webseite. o.J.

https://de.mathworks.com/help/driving/AutomatedDrivingToolbox

[15]o.V.:

Driving Scenario Designer App: Design driving scenarios, configure sensors, and generate synthetic data. Webseite. o.J.

https://de.mathworks.com/help/driving/ref/drivingscenariodesigner_app.html

[16]o.V.:

App Designer: Create apps interactively. Webseite. o.J.

https://de.mathworks.com/help/matlab/ref/appdesigner.html

[17]o.V.:

HTW Dresden K-Gebäude. Google Maps. o.J.

https://www.google.de/maps/@51.0348792,13.7392091.html

[18]o.V.:

Tuning a Multi-Object Tracker: How to tune and run a tracker to track multiple objects in the scene.Webseite. o.J.

https://de.mathworks.com/help/fusion/ug/tuning-a-multi-object-tracker.html

[19]o.V.:

Grid-Based Tracking in Urban Environments Using Multiple Lidars: How to track moving objects with multiple lidars using a grid-based tracker. Webseite. o.J.

https://de.mathworks.com/help/fusion/ug/grid-based-tracking-in-urban-environments-using-multiple-lidars.html

[20]o.V.:

trackerGridRFS: Grid-based multi-object tracker. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/trackergridrfs-system-object.html

[21]o.V.:

dbscan: Density-based spatial clustering of applications with noise. Webseite. o.J.

https://de.mathworks.com/help/stats/dbscan.html

[22]o.V.

trackingScenario: Create tracking scenario. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/trackingscenario.html

[23]o.V.

platform: Platform object belonging to tracking scenario. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/platform.html

[24]o.V.

trackingSensorConfiguration: Represent sensor configuration for tracking. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/trackingsensorconfiguration.html

[25]o.V.

theaterPlot: Plot objects, detections, and tracks in Scenario. Since R2021a. Webseite. o.J.

https://de.mathworks.com/help/radar/ref/theaterplot.html

[26]o.V.

trackPlotter: Create track plotter. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/theaterplot.trackplotter.html

[27]o.V.

getOccupancy: Get occupancy probability of locations. Webseite. o.J.

https://de.mathworks.com/help/nav/ref/occupancymap.getoccupancy.html

[28]o.V.

getEvidences: Get estimated occupancy and free evidences. Since R2021a.Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/dynamicevidentialgridmap.getevidences.html

[29]o.V.

getVelocity: Get estimated velocity and associated uncertainty. *Since R2021a.* Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/dynamicevidentialgridmap.getvelocity.html

[30]o.V.

getState: Get full estimated state and associated uncertainty. Since R2021a. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/dynamicevidentialgridmap.getstate.html

[31]o.V.

plotTrack: Plot tracks in trackingGlobeViewerSince R2021b Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/trackingglobeviewer.plottrack.html

[32]o.V.

showDynamicMap: Webseite Plot dynamic occupancy grid map. *Since R2020b*.

https://de.mathworks.com/help/fusion/ref/trackergridrfs.showdynamicmap.html

[33]o.V.

predictTracksToTime: Predict track state. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/trackergnn.predicttrackstotime.html

[34]o.V.

predictMapToTime: Webseite. Predict dynamic map to a time stamp. Since R2021a.

https://de.mathworks.com/help/fusion/ref/trackergridrfs.predictmaptotime.html

[35]o.V.

trackGOSPAMetric: Generalized optimal subpattern assignment (GOSPA) metric. Webseite. o.J.

https://de.mathworks.com/help/fusion/ref/trackgospametric-system-object.html

[36]o.V.

Introduction to Twizy. Webseite. o.J.

https://www.renaultgroup.com/en/news-on-air/news/twizy-2/

[37]o.V.

Introduction to Passat GTE. Webseite. o.J.

https://www.drivingelectric.com/volkswagen/passat/practicality

[38]o.V.

Introduction to BMW i3. PDF (online). o.J.

https://www.bmwgroup.com/ck/BMW_i3_datenblatt

[39]o.V.

Introduction to Porsche Cayenne. PDF (online). o.J.

https://pnr-prd2-pub2.newsroom.porsche.com/dam/cayenne-td-en.PDF

Eidesstattliche Erklärung

Das vorliegende Dokument wurde an der Hochschule für Technik und Wirtschaft Dresden unter der Leitung von Prof. Dr. rer. nat. Toralf Trautmann und Dipl.-Ing. Franziskus Mendt angefertigt. Hiermit erkläre ich, dass ich die vorliegende Arbeit zum Thema

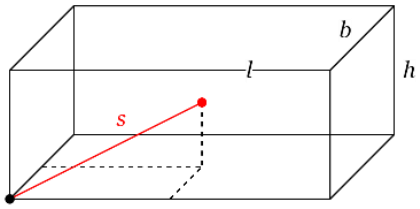
„Entwicklung eines Mehrziel-Verfolgungsalgorithmus auf Basis von LiDAR“

selbstständig und ohne Benutzung anderer Quellen und Hilfsmittel als angegeben angefertigt habe. Insbesondere versichere ich, dass ich alle wörtlichen und sinngemäßen Übernahmen aus anderen Werken als solche kenntlich gemacht habe. Ferner gestatte ich der Hochschule für Technik und Wirtschaft Dresden, die vorliegende Diplomarbeit unter Beachtung insbesondere urheber-, datenschutz-, und wettbewerbsrechtlicher Vorschriften für Lehre und Forschung zu nutzen.

Ort, Datum	Lingzhi Wu

Anhang

Anhang 3.1 Modell der Versuchsfahrzeuge



Anhang 4.1 Übersicht der Messdaten

Sensor	Szenario 1	Szenario 2	Szenario 3	Szenario 4	Szenario 5	Szenario 6
Ouster	Ouster_01	Ouster_02	Ouster_03		Ouster_05	Ouster_06
Livox	Livox_01	Livox_02	Livox_03	Livox_04	Livox_05	Livox_06
LSLid	LSLid_01	LSLid_02	LSLid_03		LSLid_05	

Anhang 4.2 Variablen über die Belegungssituation

Variable	**Beschreibung**
k	Stellt hier endgültiger Zeitindex dar.
`k_T_start`	Zeichnet die Werte des Startframes jeder Trajektorie auf.
`k_T_end`	Zeichnet die Werte des Endframes jeder Trajektorie auf.
`Num_m_occ`	Zeichnet die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0 in jedem Frame auf.
`Num_m_occ _tracks_0`	Zeichnet die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0,1 in jedem Frame auf.
`Num_m_occ _tracks_1`	Zeichnet die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0,3 in jedem Frame auf.
`Num_m_occ _tracks_2`	Zeichnet die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0,5 in jedem Frame auf.
`Num_m_occ _tracks_3`	Zeichnet die Anzahl der Gitterzellen mit einer Belegungswahrscheinlichkeit größer als 0,9 in jedem Frame auf.

Anhang 4.3 Parameter von Partikelfilter

Definition der Gitter

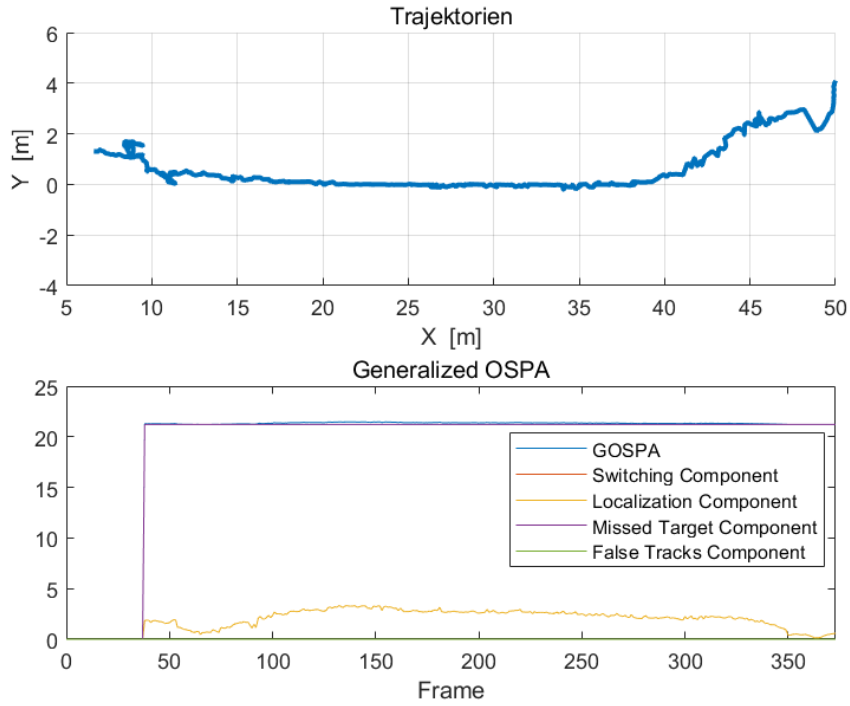
Parameter	Beschreibung	Wert
GridLength	Dimension des Gitters in x-Richtung des lokalen Koordinatensystems.	100
GridWidth	Dimension des Gitters in y-Richtung des lokalen Koordinatensystems.	100
GridResolution	Auflösung des Gitters. Sie beschreibt die Anzahl der Zellen pro Meter des Gitters in sowohl x- als auch y-Richtung.	5

Partikel

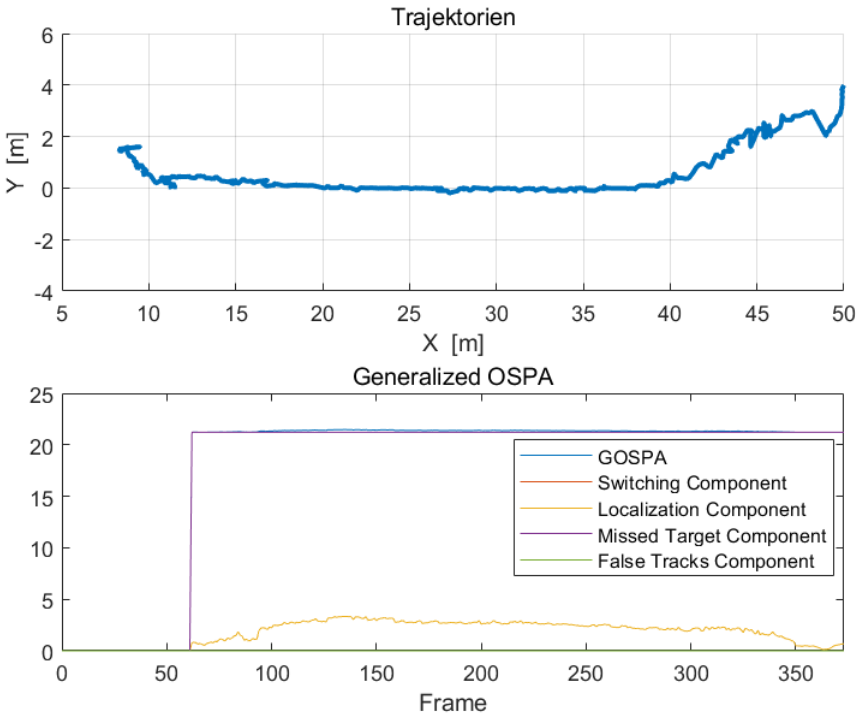
Parameter	Beschreibung	Wert
MotionModel	Bewegungsmodell für die Partikel und die Spuren	‚konstante Geschwindigkeit'
NumParticles	Anzahl der persistenten Partikel im Gitter.	1.00E+05
NumBirthParticles	Anzahl der neu erzeugten Partikel pro Schritt.	1.00E+04
BirthProbability	Die Wahrscheinlichkeit eines neu geborenen Ziels in jeder Gitterzelle.	1.00E-02
DeathRate	Die Sterberate eines Ziels pro Zeiteinheit.	1.00E-03
FreeSpaceDiscount Factor	Vertrauen in die Vorhersage des Freiraums pro Zeiteinheit.	0,8

Anhang 5.1 Simulation mit anderer Werten der Parameter

	ClusteringThreshold	MinNumCellsPerCluster	AssignmentThreshold
a	2,5	2	30
b	5	10	30



(a)



(b)

Anhang 5.2 Spezifische Punktmenge

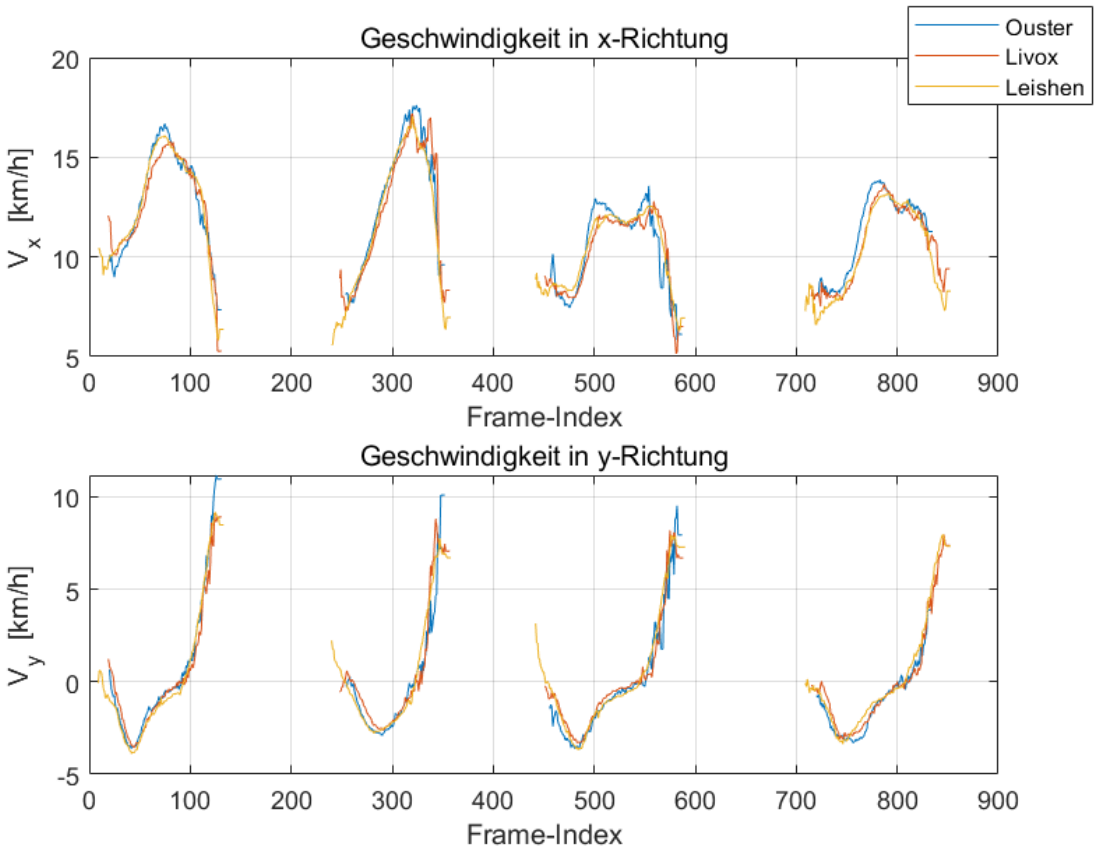
Szenario 1	Auto 1	Auto 2	Auto 3	Auto 4	Sensor
Frame	20	270	523	800	Ouster
max. Anzahl	696	1115	1093	850	
Frame	21	270	526	803	Livox
max. Anzahl	1451	2068	2186	1592	
Frame	19	270		803	LSLid
max. Anzahl	11039	18070		18697	

Szenario 2	Auto 1	Auto 2	Auto 3	Auto 4	Sensor
Frame	20	256	457	722	Ouster
max. Anzahl	754	1111	1197	861	
Frame	17	254	458	721	Livox
max. Anzahl	1382	2075	2185	1572	
Frame	18	254	455	721	LSLid
max. Anzahl	10558	17995	18272	18679	

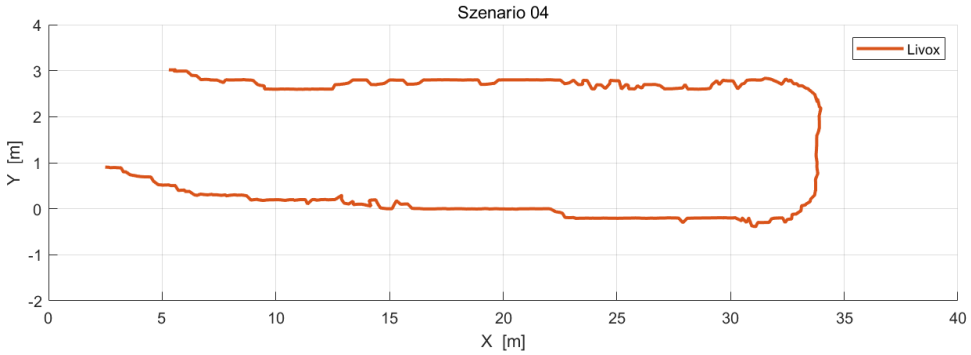
Szenario 1	Start				Ende				Sensor
	Auto 1	Auto 2	Auto 3	Auto 4	Auto 1	Auto 2	Auto 3	Auto 4	
Frame	19	263	518	793	137	384	647	904	Ouster
Anzahl	604	470	426	188	3	3	0	3	
Frame	16	259	515	793	140	387	648	906	Livox
Anzahl	884	1575	1679	554	0	0	0	6	
Frame	18	259		793	153	399		920	LSLid
Anzahl	8394	6149		7019	0	0		0	

Szenario 1	Start				Ende				Sensor
	Auto 1	Auto 2	Auto 3	Auto 4	Auto 1	Auto 2	Auto 3	Auto 4	
Frame	19	249	453	716	131	354	587	835	Ouster
Anzahl	576	406	533	208	3	3	3	8	
Frame	17	246	449	710	132	357	588	852	Livox
Anzahl	1382	1289	1128	533	3	3	3	4	
Frame	15	245	448	712	143	368	601	864	LSLid
Anzahl	5610	5727	7851	7006	0	0	0	0	

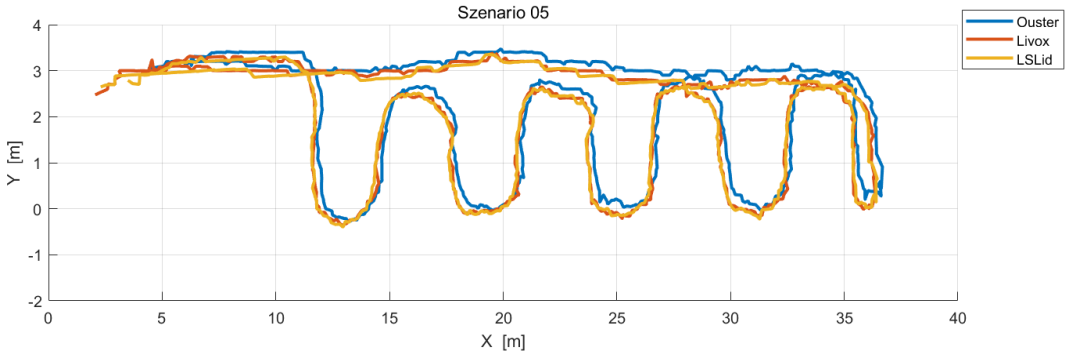
Anhang 5.3 Geschwindigkeiten in Szenario 3



Anhang 5.4 Trajektorien in Szenario 4



Anhang 5.5 Trajektorien in Szenario 5



Anhang 5.6 Analyseergebnisse des Zielverlusts in Szenario 6

```
>> Analyse_Verschwinden
 >> Ergebnis dieser Messung:

 >> Es gibt insgesamt 8 Trajektorien.

 >> Die Trajektorien des Objekts 1:

   / 1 / erscheint an Position (4.3,1.9) in Frame 35.
   ----> verschwindet an Position (38.3,2.2) in Frame 290.

   / 5 / erscheint an Position (36.2,3.1) in Frame 310.
   ----> verschwindet an Position (28.0,2.6) in Frame 365.

   / 8 / erscheint an Position (25.9,2.7) in Frame 380.
   ----> verschwindet an Position (2.1,2.6) in Frame 537.


 >> Die Trajektorien des Objekts 2:

   / 2 / erscheint an Position (3.6,2.0) in Frame 64.
   ----> verschwindet an Position (39.0,-1.2) in Frame 269.

   / 6 / erscheint an Position (37.7,-0.1) in Frame 350.
   ----> verschwindet an Position (1.8,2.6) in Frame 584.


 >> Die Trajektorien des Objekts 3:

   / 3 / erscheint an Position (6.2,5.2) in Frame 72.
   ----> verschwindet an Position (11.4,5.9) in Frame 125.

   / 4 / erscheint an Position (11.7,4.4) in Frame 235.
   ----> verschwindet an Position (18.2,1.1) in Frame 339.

   / 7 / erscheint an Position (16.3,1.8) in Frame 368.
   ----> verschwindet an Position (2.0,2.4) in Frame 480.
```

Anhang 8 Datenträger

SD_Diplomarbeit_Wu_Lingzhi_2024

- Diplomarbeit als elektronisches Dokument (.PDF)
- Messdaten_filtert
  - Livox
  - LSLid
  - Ouster
- Messdaten_Simulation
- Verfolgungsergebnisse
- Ergebnisse_Analyse
  - Analyse_Occ
  - Analyse_Abweichung_LSLid
  - Analyse_Trajektorie
  - Analyse_v
  - Analyse_yaw
  - Analyse_GOSPA
  - Analyse_Zielverlust
- Ergebnisse_Plot
- App
- Tracker_RFS
  - GridTrack_ConfigTracker_V02
  - GridTrack_Mess_Livox_V02
  - GridTrack_Mess_LSLid_V02
  - GridTrack_Mess_Ouster_V02
  - GridTrack_UpdateTracker_V02