Westfälische
Wilhelms-Universität
Münster

› Parameterbasierte Identifikation von Realdaten-Streckenabschnitten im OpenDRIVE-Format

Peter Konopatzky
Münster 2019

wissen.leben
WWU Münster

Institut für
Geoinformatik

Parameterbasierte Identifikation von Realdaten-Streckenabschnitten im OpenDRIVE-Format

Diplomarbeit

am Institut für Geoinformatik

der Westfälischen Wilhelms-Universität Münster

und am Institut für Verkehrssystemtechnik

des Deutschen Zentrums für Luft- und Raumfahrt e.V., Braunschweig

vorgelegt von

Peter Konopatzky

im Juli 2019

Erste Gutachterin: Prof. Dr. Judith Verstegen

Zweiter Gutachter: Dipl.-Geoinf. Michael Scholz

Inhaltsverzeichnis

Kurzfassung

Der Themenkomplex des automatisierten und vernetzten Fahrens wird immer aktueller und stellt die Gesellschaft vor Herausforderungen in zahlreichen Bereichen. Einer davon ist die Sicherstellung der Zuverlässigkeit von Fahrfunktionen. Die aktuell angewandte streckenkilometerbasierte Methode ist zu zeit- und ressourcenaufwändig, um im großen Maßstab für das Prüfen jeder neu entwickelten Fahrfunktion und die einhergehenden Software-Updates verwendet zu werden. Ein Lösungsansatz besteht in szenariobasierten Simulationen.

In aktuellen Projekten wird der Schwerpunkt auf die zu simulierende Verkehrsdynamik, nicht aber auf die Berücksichtigung der Umgebungsvariablen gelegt. Damit diese Simulationen nicht nur auf künstlich generierten und mathematisch-perfekt modellierten Streckenrepräsentation durchgeführt werden, sondern auch Modelle szenarienkonformer, real existierender Strecken als Simulationsgrundlage genutzt werden können, untersucht diese Arbeit, unter welchen Umständen sich Realdaten-Streckenabschnitte parameterbasiert und automatisch identifizieren lassen.

Für diesen Zweck ist ein Algorithmus entwickelt und implementiert worden, der in OpenDRIVE-Daten deutscher Autobahnen nach Streckenabschnitten sucht, die den Vorgaben von Szenarien aus dem Kontext des Automotive-Forschungsprojekts PEGASUS entsprechen.

Eine solche Streckenidentifikation stellt sich als durchführbar heraus, aber der zu betreibende Aufwand ist stark realdatenformat- und parameterabhängig. In einem Zwischenschritt des Identifikationsvorgangs fallen Daten an, anhand derer einfach festzustellen ist, an welcher Stelle des realen Streckenverlaufs sich bestimmte Eigenschaften der Strecke ändern. Diese Zwischenergebnisse können als Grundlage für eine umfassende Parametrisierungen von Vermessungsdaten dienen.

Danksagung

In erster Linie möchte ich Michael Scholz danken, der mir als Betreuer bei Problemen aller Art zur Seite stand. Dies gilt insbesondere für die Endphase beim Verfassen dieser Arbeit. Gleichzeitig gebührt Frau Verstegen eine ordentliche Portion Dank für ihre strukturgebenden Hinweise und, vor allem, ihr Verständnis.

Nicht weniger Dank geht an Mirjam, Sven und Tina für das Korrekturlesen, die Unterstützung in den letzten Wochen (insbesondere durch Strom und Tee) und den LaTeX-Support. Ihr wart eine große Hilfe!

Einen nicht unerheblichen Beitrag zum Entstehen dieser Arbeit hat außerdem die Londoner Band *DragonForce* geleistet, zu deren Musik sich besonders effizient arbeiten lässt. Thank you, guys!

1 Einleitung

1.1 Motivation

Automatisiertes und vernetztes[1] Fahren ist ein zeitgenössisches Thema, welches neben ethischen und rechtlichen Fragestellungen [1] zahlreiche technische Herausforderungen mit sich bringt; vor allem in den Bereichen Security [2] und Safety. Hersteller und Zulassungsstellen müssen sich die Frage nach der Zuverlässigkeit einer Fahrfunktion stellen. Wie sicher muss die Fahrfunktion sein? Und wie kann sichergestellt werden, dass sie ihre Aufgabe ausreichend gut und zuverlässig erfüllt? [3]

Legt man den Anspruch zugrunde, eine Fahrfunktion solle eine geringere Unfallwahrscheinlichkeit als menschliche Fahrzeugführer aufweisen, wird klar, dass das ausschließliche Testen im realen Straßenverkehr keine zufriedenstellende Lösung ist. Menschliche Fahrzeugführer verursachen etwa alle 600 Millionen Fahrkilometer einen Unfall mit Todesfolge. Um mit 95%iger Wahrscheinlichkeir sagen zu können, dass eine Fahrfunktion sicherer als ein Mensch fährt, müsste sie daher gut sechs Milliarden Fahrkilometer ohne tödliche Unfälle hinter sich bringen – und zwar nach jeder Modifikation und Weiterentwicklung. Die Nachteile dieser entfernungsbasierten Methode sind zahlreich. Sie ist kosten-, material- und zeitintensiv, schlecht reproduzierbar und gefährdet nicht zuletzt andere Verkehrsteilnehmer. [3, 4]

Mit szenariobasierten Computersimulationen kann ein Großteil dieser Fahrkilometer virtuell und zielgerichtet absolviert werden, bevor auf Teststrecken und schließlich im Straßenverkehr getestet wird. Simulationen sind finanziell günstiger sowie schneller,

[1] Fahrzeug-zu-Fahrzeug-Kommunikation (*Car-to-Car* oder *Vehicle-to-Vehicle* (C2C, V2V)) und Fahrzeug-Infrastruktur-Kommuniktion (C2I, V2I), etwa zur Stauwarnung oder Verkehrsflussoptimierung.

kontrolliert und gefahrlos durchzuführen [5]. Diese Vorteile kommen allerdings nur zum Tragen, wenn die zugrundeliegenden Szenarien von herausragender Qualität sind.

Die Qualität wird vor allem durch den Realismus der Simulationen bestimmt. In die Optimierung der Verkehrsfluss- und -maneuversimulation wird bereits viel Arbeit gesteckt. Der Realismus der simulierten Umgebung, einschließlich der Eigenheiten der Straße, werden vernachlässligt. Das Ziel dieser Arbeit ist ein Beitrag zur Aufwertung von Simulationen durch realistischere Umgebungen.

1.2 Fragestellung

Das Simulieren des Verhaltens einer Fahrfunktionen in Szenarien ist ein wesentlicher Bestandteil ihres Validierungsprozesses [6]. Die Anzahl zu simulierender Manöver mag niedrig sein, aber die daraus ableitbaren konkreten Situationen werden unzählbar. Die möglichen Manöver eines Spurhalteassistenten können etwa auf einen Bremsvorgang des Vordermanns und das Einscheren eines dritten Fahrzeugs heruntergebrochen werden. Die zu testenden Situationen können sich jedoch in Details wie den absoluten und relativen Geschwindigkeiten aller beteiligten Fahrzeuge, der Abruptheit des Einschervorgangs, dem Gewicht des simulierten Fahrzeugs, den Ausweichmöglichkeiten, wetterbedingten Sichtweiten und Bodenverhältnissen oder den geometrischen Eigenschaften der Straße unterscheiden. Es gibt unendlich viele Kombinationen von Ausprägungen dieser Parameter.

Minnerup [7] kümmert sich um die Bestimmung sinnvollerweise zu testender Parameterkombinationen, aber konzentriert sich allerdings auf die Verkehrssituation. Das Automotive-Forschungsprojekt PEGASUS[2] nutzt mathematisch perfekte, idealtypische Strecken und variiert im Wesentlichen nur die Verkehrssituation. Die Eigenheiten der Straße werden kaum variiert. Die virtuelle Verwendung real existierender Streckenabschnitte, welche den in PEGASUS entwickelten Vorgaben entsprechen, aber in ihren Eigenheiten nicht systematisch künstlich generiert werden könnten, würde diese Lücke zu füllen beginnen.

[2] Projekt zur Etablierung von generell akzeptierten Gütekriterien, Werkzeugen und Methoden sowie Szenarien und Situationen zur Freigabe hochautomatisierter Fahrfunktionen, siehe [8].

Daraus ergibt sich die Forschungsfrage: *Unter welchen Umständen lassen sich Streckenabschnitte, die den Vorgaben von Simulationsszenarien entsprechen, automatisiert in Daten realer Straßennetze identifizieren?* Untersucht wird diese Fragestellung anhand von Daten eines Abschnitts einer deutschen Autobahn, welche im OpenDRIVE-Format vorliegen. Für diesen Zweck wird eine Suchfunktion implementiert, welche sich später in den Validierungsprozess einbinden ließe, der im Rahmen des Forschungsprojekts PEGASUS entwickelt wurde.

1.3 Gliederung dieser Arbeit

Die vorliegende Arbeit gliedert sich in sechs Kapitel. Nach der Einleitung werden in Kapitel 2 die Grundlagen für das Verständnis dieser Arbeit gelegt. Es werden den Anwendungsfall betreffene Hintergrundinformationen und themenverwandte Arbeiten vorgestellt.

In Kapitel 3 wird die Methodik dieser Arbeit ausgeführt. Im anschließenden Kapitel 4 werden Ergebnisse vorgestellt und interpretiert. Abschließend diskutiert Kapitel 5 die Ergebnisse, stellt die Einschränkungen dieser Arbeit vor und gibt einen Ausblick über mögliche Anknüpfungspunkte.

2 Grundlagen

In diesem Kapitel werden die Grundlagen für das Verständnis dieser Arbeit gelegt. Da der betrachtete Anwendungsfall (Kapitel 3.1) auf Realdaten deutscher Autobahnen (2.1) im Dateiformat OpenDRIVE (2.3) basiert und im Kontext szenarienbasierter Simulationen (2.2) und des Forschungsprojekts PEGASUS (2.4) steht, werden diese Themen in eigenen Unterkapiteln besprochen. Weitere Abschnitte gehen außerdem auf Verfahren für den Vergleich linienbasierter Geometrien (Kapitel 2.5) und die verwendeten Programmbibliotheken ein (Kapitel 2.6).

2.1 Entwurf von Autobahnen

In Deutschland unterliegt die Anlage von Straßen Vorgaben, welche von Expertengruppen erstellt und als technische Regelwerke veröffentlicht werden. Das aktuell für die Anlage von Autobahnen gültige Regelwerk heißt *Richtlinien für die Anlage von Autobahnen* (RAA) und wurde 2008 herausgegeben [9]. Enthalten sind Vorgaben, denen beim Bau beziehungsweise der Erneuerung bestehender Autobahnen gefolgt und von denen nur in begründeten Ausnahmefällen abgewichen werden soll. Das Regelwerk ersetzt unter anderem die autobahnbezogenen Teile der Vorgängerregelwerke *Richtlinien für die Anlage von Straßen, Teil: Linienführung* (von 1999) und *Richtlinien für die Anlage von Straßen, Teil: Linienführung* (von 1996).

Der Entwicklung der RAA lagen verschiedene Anforderungen zugrunde. Beispielsweise sollen Autobahnen nicht nur für bestimmte Geschwindigkeiten und Verkehrsaufkommen geeignet sein, sondern Sicherheitsanforderugen erfüllen, die Umgebung schützen und möglichst kostengünstig in Errichtung, Betrieb und Wartung sein.

Es wird zwischen den drei Entwurfsklassen Fern-/Überregionalautobahnen (EKA 1),

autobahnähnliche Straßen (EKA 2) und Stadtautobahnen (EKA 3) unterschieden. Die erste Entwurfsklasse differenziert zusätzlich in Fernautobahnen (EKA 1 A) und Überregionalautobahnen (EKA 1 B). Entwurfsklassenabhängig gelten unterschiedliche Vorgaben.

So existieren Vorgaben für den Straßenquerschnitt, welche etwa die Anzahl und Breite der Fahrstreifen betreffen. Diese Vorgaben werden *Regelquerschnitte* genannt. Für jede Entwurfsklasse existiert genau ein Regelquerschnitt je möglicher Fahrstreifenanzahl; zuzüglich spezieller Regelquerschnitte für Brücken- und Tunnelabschnitte. Außerdem werden entwurfsklassenabhängige Empfehlungen gegeben, beispielsweise für den Abstand zwischen Anschlussstellen oder die geometrische Ausgestaltung von Kurven. So wird als Mindestabstand zwischen zwei Anschlusstellen von Autobahnen der ersten Entwurfsklasse ein Mindestabstand von $5\,000\,\mathrm{m}$ (EKA 1 B) beziehungsweise $8\,000\,\mathrm{m}$ (EKA 1 A) empfohlen [9, S. 17].

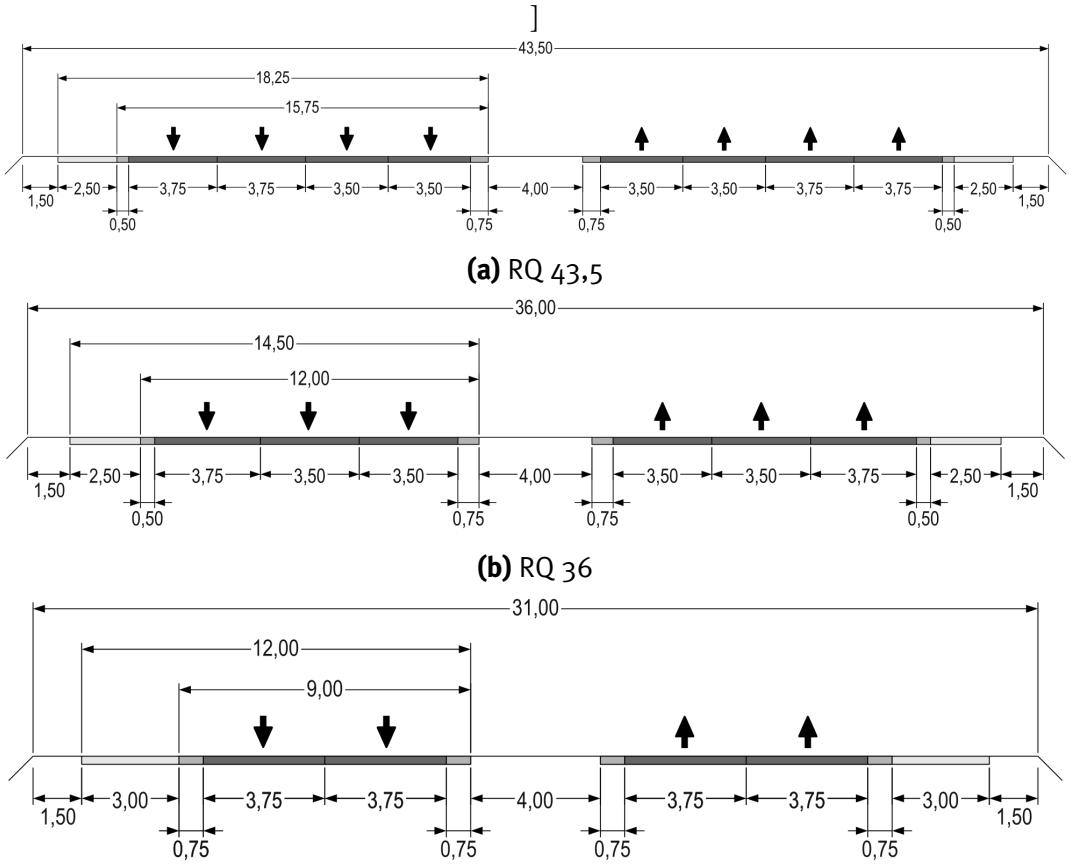
Im Folgenden werden für die untersuchte Fragestellung relevante Aspekte der Richtlinien für die Anlage von Autobahnen vorgestellt. Das umfasst die Regelquerschnitte der ersten Entwurfsklasse sowie die grundlegenden Vorgaben für Trassierungselemente, welche den Straßenverlauf maßgeblich bestimmen.

2.1.1 Regelquerschnitte

Ein Regelquerschnitt beschreibt den Sollquerschnitt einer Straße. Unter anderem legt er Anzahl und Breite von Fahr- und Standstreifen fest. Die Benennung von Regelquerschnitten richtet sich nach der Gesamtbreite des bebauten Bereichs. Die Auswahl des Regelquerschnitts beim Bau einer Autobahn richtet sich unter anderem nach der erwarteten Verkehrsstärke, wie in Tabelle 2.1 aufgeschlüsselt wird.

Für diese Arbeit sind die Standard-Regelquerschnitte der ersten Entwurfsklasse von Interesse: *RQ 43,5* für vierstreifige, *RQ 36* für dreistreifige und *RQ 31* für zweistreifige Fern- bzw. Überregionalautobahnen. Darhüber hinaus sind Regelquerschnitte der Entwurfsklassen EKA 2 und EKA 3 sowie die Spezial-Regelabschnitte für Brücken- und Tunnelabschnitte in den RAA enthalten.

[1] Grafiken von Wikipedia-Autor Nils Gruschka, abgerufen jeweils am 3. Juli 2019:



(a) RQ 43,5

(b) RQ 36

(c) RQ 31

Abbildung 2.1: Die drei EKA-1-Regelquerschnitte (nach [9]). Fahrtrichtungsangabe in schwarz. Fahrstreifen in dunkelgrau, Bankett in grau, Standstreifen in hellgrau. Angaben in Meter.[1]

Tabelle 2.1: Fahrstreifenanzahl und Regelquerschnitt in Abhängigkeit des durchschnittlichen Verkehrsaufkommens, gemessen in Tausend Fahrzeugen je Tag (nach [9, S. 23])

Verkehrsstärke	Fahrstreifen	Regelquerschnitt
18–68	2×2	RQ 31
62–102	2×3	RQ 36
92–120	2×4	RQ 43,5

Abbildung 2.1 skizziert die drei EKA-1-Regelquerschnitte. Auffallend sind die unterschiedlichen Breitenvorgaben für die Fahrstreifen, welche sich aus der Nutzung durch Schwerlastverkehr ergeben. Die äußeren Fahrstreifen, welche der Schwerlastverkehr überwiegend nutzt, sollen aus Sicherheitsgründen $3{,}75\,\mathrm{m}$ messen, wohingegen für die inneren Fahrstreifen eine Breite von $3{,}50\,\mathrm{m}$ als ausreichend erachtet und daher vorgegeben wird. So sollen Platzbedarf und Konstruktionskosten reduziert werden [9, S. 20]. Daher sind Breitenangaben für Fahrstreifen im Gegensatz zu den erwähnten Abstandsempfehlungen nicht als Mindestangaben, sondern als präzise Sollwerte zu verstehen.

2.1.2 Trassierungselemente

Der Verlauf eines Verkehrswegs, ob geplant oder umgesetzt, wird als *Trasse* bezeichnet, welche wiederum aus *Trassierungselementen* (auch: *Entwurfselemente*) besteht. Nicht alle Arten von Trassierungselementen beeinflussen den geometrischen Verlauf eines Verkehrswegs, aber genau jene sind für diese Arbeit von Interesse. Bei geometrischen Trassierungselementen, welche den Straßenverlauf beschreiben, handelt es sich typischerweise um Geraden, Kreis- und Übergangsbögen. Damit beim Übergang von einer Geraden in eine enge Kurve nicht schlagartig eingelenkt werden muss, werden Geraden und Kreisbögen üblicherweise mit einem Übergangsbogen (siehe Abbildung 2.2) verbun-

RQ 43,5: https://de.wikipedia.org/w/index.php?curid=4873803,
RQ 36: https://commons.wikimedia.org/w/index.php?curid=62424646,
RQ 31: https://de.wikipedia.org/w/index.php?curid=4886867

den, der eine stetige Krümmungsänderung aufweist. Für diesen Zweck bieten sich etwa Spiralbögen an.

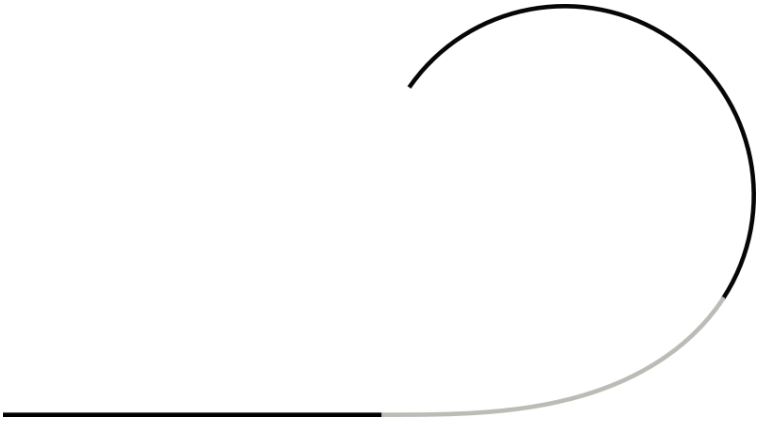


Abbildung 2.2: Drei Trassierungselemente: ein Übergangsbogen mit sich stetig ändernder Krümmung (grau) zwischen Gerade und Kreisbogen (jeweils schwarz)

Die Richtlinien geben folgende Empfehlungen für Geraden. Zwischen zwei identisch orientierten Kurven verlaufende Geraden sollten eine Länge von mindestens 400 m aufweisen, um von Fahrzeugführern als eigenes Entwurfselement wahrgenommen zu werden. Außerdem sollten Geraden eine Länge von 2 000 m nicht überschreiten, weil monotone Abschnitte ermüdend wirken, Geraden zu schnellerem Fahren verleiten und das Abschätzen von Entfernungen und Geschwindigkeiten erschweren. [9, S. 28]

Für Kurven gelten entwurfsklassenabhängige Vorgaben, welche vor allem durch die entsprechenden Richtgeschwindigkeiten und damit einhergehenden notwendigen Sichtweiten zustande kommen. Engere Kurven sind nicht nur schwieriger zu durchfahren, sondern verringern auch die Sichtweite, etwa durch Bewuchs des Mittelstreifens. Um hektisches Lenken zu vermeiden, werden Minimallängen für Kurven vorgeschrieben. Tabelle 2.2 bietet einen Überblick. Einen Sonderfall bilden Kurven, die auf Geraden ohne Geschwindigkeitslimit und mit einer Mindestlänge von 500 m folgen. Hier beträgt der Empfehlung für den Mindestradius einer Kurve 1 300 m [9, S. 29].

Alterantiv zum Kurvenradius lässt sich die Enge eines Kurvenverlaufs durch die äqui-

Tabelle 2.2: Richtgeschwindigkeit, Vorgaben für Kurvenradius und -länge nach Entwurfsklasse (aus [9, S. 28]).

Entwurfsklasse	Richtgeschwindigkeit	Mindestradius	Mindestlänge
EKA 1 A	130 km/h	900 m	75 m
EKA 1 B	120 km/h	720 m	
EKA 2	100 km/h	470 m	55 m
EKA 3	80 km/h	280 m	

valente Größe der *Kurvenkrümmung* ausdrücken. Der Kurvenradius entspricht dem Radius des imaginären Kreises, dessen Krümmung der des Streckenverlaufs entspricht. Die Kurvenkrümmung ist definiert als der Kehrwert des Kurvenradius. Ein geringerer Kurvenradius führt zu stärkerer Krümmung, also einer „schärferen“ Kurve. Beide Größen lassen sich mit Formel 2.1 ineinander umwandeln. Demnach entspricht ein Kurvenradius von beispielsweise $250\,\mathrm{m}$ einer Kurvenkrümmung von $1/250 = 0{,}004\,\mathrm{m}^{-1}$.

$$\text{Kurvenkrümmung} = 1/\text{Kurvenradius} \tag{2.1}$$

2.2 Der Szenario-Begriff

Wachenfeld und Winner [4] zeigen, dass streckenbasierte Zulassungsprozesse enorme Anforderungen an die Menge zu fahrender Streckenkilometer. Anschaulich wird das anhand der in der Einleitung erwähnten Größenordnung von mehreren Milliarden Kilometern, die eine Fahrfunktion nach jeder größeren Veränderung am Code absovieren müsste. Das ist nicht zuletzt als unwirtschaftlich zu bezeichnen Ein Alternativvorschlag, den auch Schuldt [6] vorbringt, sind szenarienbasierte Simulationen.

Zahlreiche Forscher im Automotive-Bereich nutzen den Szenariobegriff, definieren ihn allerdings nicht einheitlich und erschweren dadurch vergleichende Arbeiten. Bagschick et al. [10] schlagen daher Definitionen vor, welche auf der Arbeit von Ulbrich et al. [11] basieren. Diese und die Umsetzung im Rahmen des Forschungsprojekts PEGASUS, welches Teil des Anwendungsfalls dieser Arbeit ist (siehe Kapitel 3.1), werden im Folgenden vorgestellt.

2.2.1 Abstraktionsebenen

Bagschick et al. [10] definieren drei Abstraktionsebenen für Szenarien: *funktionale*, *logische* und *konkrete* Szenarien.

> „Funktionale Szenarien stellen Betriebsszenarien des Entwicklungsgegenstands auf semantischer Ebene dar. Die Entitäten und Beziehungen zwischen den Entitäten der Anwendungsdomäne werden in sprachlich gefassten Szenarien ausgedrückt." [10, Kap. 3.1]

Funktionale Szenarien beruhen auf Beschreibungen in natürlicher Sprache. Ein beispielhaftes Szenario könnte durch die Punkte *Fernautobahn in enger Linkskurve*, *zähfließender Verkehr* und *sommerliches Wetter* beschrieben werden.

> „Logische Szenarien stellen Betriebsszenarien durch Entitäten und Beziehungen dieser Entitäten mithilfe von Parameterbereichen im Zustandsraum dar. Für die einzelnen Parameterbereiche können optional statistische Verteilungen angegeben werden." [10, Kap. 3.2]

Logische Szenarien spezifizieren das Szenario näher, indem die sprachliche Beschreibung in Parameter und Wertebereiche umgewandelt wird. Unser Szenario könnte etwa durch die Parameter *Regelquerschnitt*, *Kurvenradius*, *Staugeschwindigkeit* und *Temperatur* sowie die Wertebereichen $\{\text{RQ } 31, \text{RQ } 33, \text{RQ } 42{,}5\}$, $1{,}0\text{–}2{,}0\,\mathrm{km}$, $7{,}0\text{–}20{,}0\,\mathrm{km/h}$ und $15\text{–}40\,^{\circ}\mathrm{C}$ beschrieben werden.

> „Konkrete Szenarien stellen Betriebsszenarien eindeutig durch Entitäten und Beziehungen dieser Entitäten mithilfe von festen Werten im Zustandsraum dar." [10, Kap. 3.3]

Auf der dritten Abstraktionsebene, den konkreten Szenarien, erhalten die Parameter des logischen Szenarios einen konkreten Wert aus ihrem Wertebereich. So wird das Beispielszenario durch den *Regelquerschnitt 36*, den Kurvenradius $1{,}5\,\mathrm{km}$, die Staugeschwindigkeit $16\,\mathrm{km/h}$ und die Lufttemperatur $27\,^{\circ}\mathrm{C}$ konkretisiert. Eine Übersicht der drei Abstraktionsebenen gibt Tabelle 2.3.

Tabelle 2.3: Unterschiedliche Abstraktionsebenen eines Szenarios (angelehnt an [10, 12]).

funktional	logisch		konkret
Fernautobahn	Regelquerschnitt:	RQ 33, 36 oder 43,5	RQ 33
Linkskurve	Kurvenradius:	1,0–2,0 km	1,5 km
zähfließender Verkehr	Geschwindigkeit:	7–20 km/h	16 km/h
Sommer	Lufttemperatur:	15–40 °C	27 °C

2.2.2 Beschreibungsebenen

Szenarien werden in Beschreibungsebenen gegliedert. In der Literatur finden sich Szenariendefinitionen mit unterschiedlich vielen Beschreibungsebenen. So schlägt Schuldt [6] ein vierstufiges Modell vor, wohingegen PEGASUS anfangs mit einem fünfstufigen Modell plante [12], letztlich aber ein sechsstufiges verwendet [13].

Die erste Beschreibungsebene umfasst Geometrie und sonstige Straßeneigenschaften (etwa Querschnitt und Länge), die zweite die Leitinfrastruktur (etwa Leitplanken und Schilderbrücken) und die dritte beschreibt vorübergehende Änderungen der ersten beiden Ebenen (etwa Baustellenbeschilderungen). In der vierten Beschreibungsebenen werden Verkehrssituation und -dynamik (etwa Geschwindigkeiten und Manöver), in der fünften Umweltbedingungen (etwa Temperatur und Niederschlag) und in der sechsten digitale Informationen (etwa Staumeldungen und C2X-Kommunikation) beschrieben. Eine Skizze der im Rahmen von PEGASUS verwendeten Beschreibungsebenen findet sich in Abbildung 2.3.

Ein für die Repräsentation der ersten drei Beschreibungsebenen nutzbares Dateiformat ist OpenDRIVE, welches auch im Rahmen von PEGASUS genutzt wird. Da der Anwendungsfall dieser Arbeit die ersten beiden Ebenen umfasst, wird OpenDRIVE im nächsten Unterkapitel beschrieben.

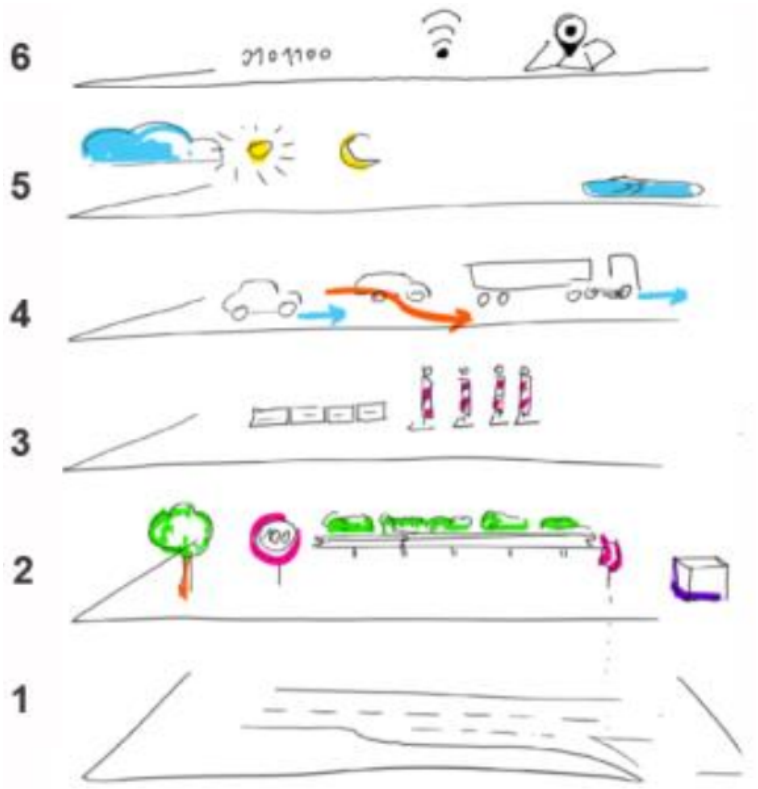


Abbildung 2.3: Skizze des 6-Ebenen-Modells zur Beschreibung von Szenarien (nach [13]): Straßenebene (Ebene 1), Leitinfrastruktur (Ebene 2), vorübergehende Änderungen der ersten beiden Ebenen (Ebene 3), Verkehr (Ebene 4), Umwelt (Ebene 5), digitale Informationen (Ebene 6).

2.3 OpenDRIVE

OpenDRIVE[2] ist ein offenes, XML-basiertes Dateiformat zur detaillierten und hochgenauen Beschreibung von Straßen. Anfang 2019 wurde die aktuelle Spezifikation 1.5 veröffentlicht. Die dieser Arbeit zugrundeliegenden Daten liegen in Version 1.4H [14] vor, welche ein De-facto-Standard im Fahrsimulationsbereich ist. In Zukunft soll der Verein *ASAM e.V.*[3] die Weiterentwicklung des Standards übernehmen und OpenDRIVE in einen Industriesstandard überführen.

Ursprünglich entworfen wurde OpenDRIVE zur Verwendung in Fahrsimulationen, bei denen Menschen in Fahrzeugsimulatoren sitzen und auf Bildschirmen eine künstliche virtuelle Umgebung aus ihrer Perspektive sehen. Inzwischen werden OpenDRIVE-Daten allerdings auch aus Kataster- oder Multisensordaten generiert [15, 16], zur Darstellung komplexer realer Strecken genutzt [17] und als zusätzliche Wissensbasis für Assistenzfunktionen in Fahrzeugen verwendet. Generell wird OpenDRIVE im Automotive-Bereich immer verbreiteter und bildet die Streckenbasis in großen Testfeldern in Düsseldorf[4], Niedersachsen[5,6], Bayern[7], Österreich[8] und China[9]. Obwohl OpenDRIVE in den Spezifikationen 1.4 und 1.5 nicht explizit auf die Abbildung realer Geodaten ausgelegt ist, bietet es ausreichend Flexibilität für diesen Zweck [17].

Die nächsten Unterkapitel befassen sich mit der Format-Spezifikation von OpenDRIVE 1.4. Neben der grundlegenden Struktur (Kapitel 2.3.1 werden Probleme (Kapitel 2.3.2) und Weiterentwicklungen (Kapitel 2.3.4 und 2.3.5) angesprochen.

[2] http://www.opendrive.org
[3] Association for Standardization of Automation and Measuring Systems.
[4] https://www.komod-testfeld.org
[5] https://www.dlr.de/ts/aim
[6] https://www.dlr.de/dlr/desktopdefault.aspx/tabid-10081/151_read-33501/
[7] https://www.bmvi.de/DE/Themen/Digitales/Digitale-Testfelder/Digitale-Testfelder.html
[8] https://www.alp-lab.at/
[9] http://www.jiemian.com/article/1638467.html

2.3.1 Struktur

OpenDRIVE beschreibt eine Datenbank verschiedener straßenrelevanter und miteinander verknüpfter Elemente. Hierfür wird eine XML[10]-Struktur verwendet.

Eine OpenDRIVE-Datei beginnt mit einem `<header>`-Element, welches Informationen über die Versionsnummer, den Zeitpunkt der Erstellung, den Ersteller und möglicherweise eine Georeferenzierung enthält. Es folgt mindestens ein Straßenelement (`<road>`) und beliebig viele Kreuzungselemente (`<junction>`). Die Verknüpfung von Straßenelementen untereinander oder mit Kreuzungselementen geschieht über `id`-Attribute. Straßenelemente können je ein weiteres Straßen- oder Kreuzungselement als Vorgänger und Nachfolger haben. Verzweigungen und Kreuzungen stellen Spezialfälle dar, die eine gesonderte Behandlung erfordern und im Rahmen dieser Arbeit, welche sich auf anschlussstellenfreie Autobahnabschnitte beschränkt (siehe Kapitel 3.1), außen vor gelassen werden.

OpenDRIVE beschreibt ein Straßennetz mit seinem geometrischen Verlauf auf stete, kontinuierliche Weise. Der Verlauf eines `<road>`-Elements wird, ausgehend von einem Startpunkt, mit einer Folge geometrischer Grundformen definiert. Dies können Geraden, Polynomfunktionen, Kreisbogen- oder Spiralsegmente sein (siehe Abbildung 2.4); ganz ähnlich wie Autobahnverläufe in den Richtlinien zur Anlage von Autobahnen mathematisch modelliert werden (siehe Abbildung 2.2). Der Startpunkt wird in dem im `<header>`-Element genutzten Koordinatensystem angegeben, könnte allerdings auch aus dem Ende des Vorgänger-Elements rekonstruiert werden. Dieser Verlauf wird *Referenzlinie*, *Track* oder *lane zero* genannt.

Im Rahmen von OpenDRIVE finden zwei Referenzsysteme Verwendung. Zum Einen ein zweidimensional-kartesisches, welches einem realen projizierten Koordinatenreferenzsystem (ohne Höhenangaben) entsprechen kann und dessen Koordinaten im Format (x, y) angegeben werden. Zum anderen gibt es das Track-Referenzsystem. Hierbei handelt es sich um lineare Referenzierung entlang der Referenzlinie eines Straßenelements. Dieses wird für relative Verschiebungen von Staßenverläufen oder die Position von Objekten in Relation zur Referenzlinie genutzt. Koordinaten werden als (s, t, h)-

[10] Extensible Markup Language.

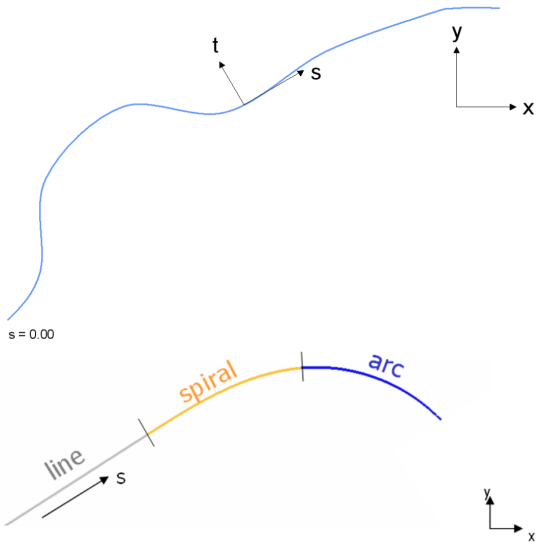


Abbildung 2.4: Die beiden grundlegenden OpenDRIVE-Koordinatensysteme und die Referenzlinie eines `<road>`-Elements, zusammengesetzt aus drei unterschiedlichen Geometrien (aus [14]).

Werte angegeben, wobei s die Position entlang der Referenzlinie, t die horizontale und h die vertikale Abweichung von dieser bezeichnet. Das Zusammenspiel der beiden Koordinatensysteme wird in Abbildung2.4 verdeutlicht.

Ein Straßenelement enhält wiederum weitere Elemente, von denen einige optional und einige obligatorisch sind. An dieser Stelle sollen nur die für den Kontext dieser Arbeit relevanten Elemente vorgestellt werden. Das wichtigste Element ist das `<planView>`-Element, das den Trassenverlauf des Straßenelements beschreibt. Es besteht aus Geometrielementen (`<geometry>`), welche aufeinanderfolgende Abschnitte des Trassenverlaufs beschreiben. Die Position und Orientierung dieser Geometrien wird im (s, t, h)-Referenzsystem angegeben und auf ihrer Basis werden später die Parameter Straßenorientierung und Kurvenradius bestimmt (siehe Kapitel 3.2.1).

Eine weiterer notwendiger Bestandteil eines jeden Straßenelements ist das `<lanes>`-Element, welches wiederum aus `<laneSection>`- und `<lane>`-Elementen besteht. Diese Elemente unterteilen das Straßenelement in Abschnitte und definieren je Ab-

schnitt die Anzahl und Art von Fahrstreifen, den Verlauf der Fahrstreifengrenzen, assoziierter Fahrbahnmarkierungen (`<roadMark>`) oder sonstiger, individuell definierter Objekte (`<userData>`) in Relation zur Referenzlinie. Auf Basis dieser Angaben wird im Rahmen dieser Arbeit der der Autobahn zugrundeliegende Regelquerschnitt abgeleitet (Kapitel 3.2.1).

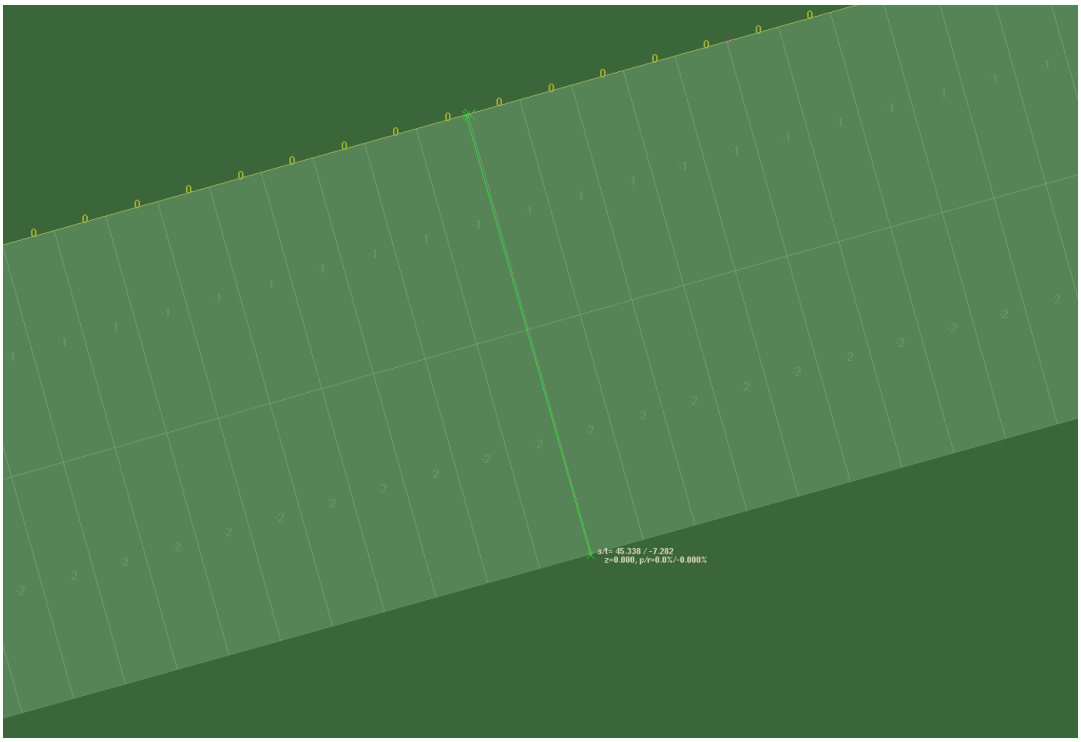
Einzelne Fahrstreifen sind durchnummeriert. Die Mitte der Fahrbahn oder der linke Rand des innersten Fahrstreifen werden als breitenloser „nullter“ Fahrstreifen (*lane zero*) betrachtet. Mit ihm soll die Referenzlinie üblicherweise identisch beziehungsweise deckungsgleich sein. In der Praxis kann die Referenzlinie jedoch durch `<laneOffset>`-Elemente von dieser Vorgabe abweichen, wie in Abbildung 2.5 beispielhaft zu erkennen ist. Die tatsächlichen Fahrstreifen in Fahrtrichtung werden mit -1 (innerster Fahrstreifen), -2 und so weiter nach außen nummeriert. Die Fahrstreifen in Gegenrichtung hingegen werden mit positiven `id`-Attributwerten (1, 2, ...) belegt, sofern sie nicht in Bezug auf eine eigene Referenzlinie mit umgekehrter Richtung definiert sind.

Wie bereits beschrieben werden geometrische Formen nicht durch absolute Ortsangaben, sondern durch mathematische Funkionen spezifiziert. Die absoluten `(x,y)`-Koordinaten der Referenzlinie werden durch eine Funktion der Position entlang der Referenzlinie angegeben, deren Parameter als Attribute im entsprechenden `<geometry>`-Element angegeben sind. Ähnliches gilt beispielsweise auch für Fahrstreifenbreiten, die ebenfalls als kubische Funktion der Position entlang der Referenzlinie angegeben werden. Ein Beispiel lässt sich in Listing 2.1 betrachten. Die zugehörige Formel für die Fahrstreifenbreite an der Stelle ds auf der Referenzlinie lautet wie folgt [14, S. 55]:

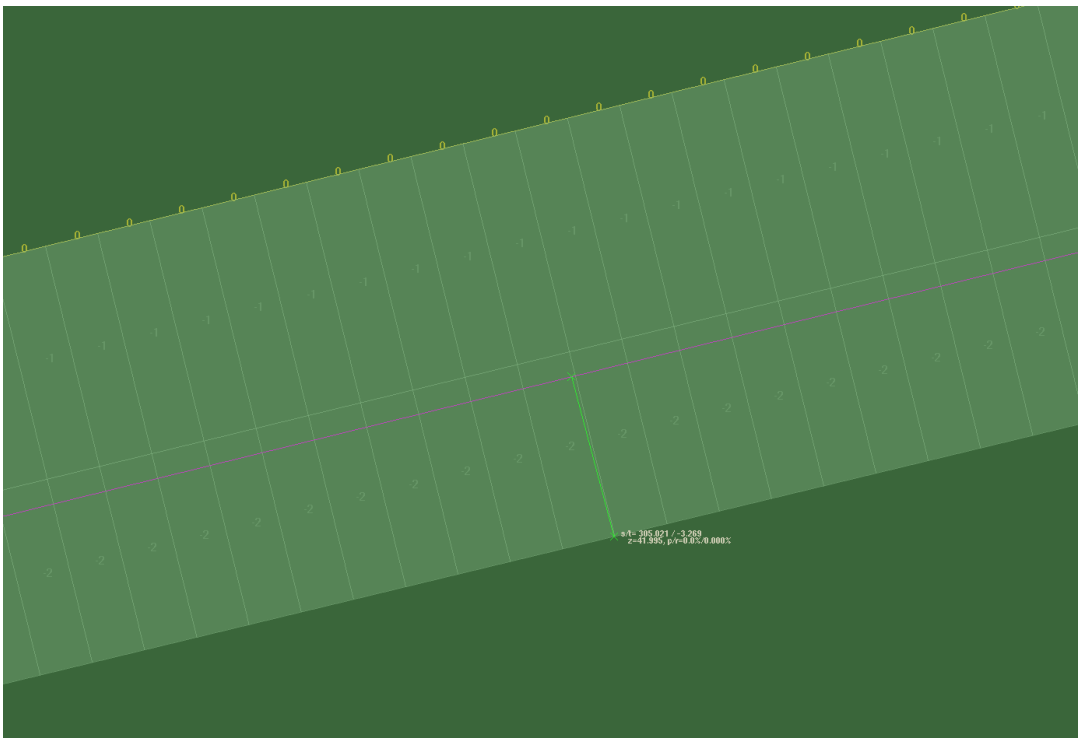
$$\text{width} = a + b \cdot ds + c \cdot ds^2 + d \cdot ds^3$$

Listing 2.1: Fahrstreifenbreite, angegeben mittels `<width>`-Element.

```
1 <lane id="-1" type="driving" level="false">
2   <width sOffset="0.0" a="3.0" b="0.0" c="0.0" d="0.0"/>
3 </lane>
4
```



(a) Referenzlinie deckungsgleich mit (und verdeckt von) *lane zero* (Linie und Beschriftung gelb)



(b) nicht deckungsgleich

Abbildung 2.5: Zwei Beispielverläufe der Referenzlinie (lila) einer zweistreifigen Fahrbahn. Die hellgrüne Linie ist die kürzeste Verbindung zwischen Mauszeigerposition und Referenzlinie – eine Funktion eines kommerziellen OpenDRIVE-Editors.

```
<lane id="-2" type="shoulder" level="false">
  <width sOffset="0.0" a="2.0" b="0.0" c="0.0" d="0.0"/>
</lane>
```

2.3.2 Flexibilität

Nicht nur die Lage der Referenzlinie stellt sich in der Praxis als flexibel heraus. OpenDRIVE bietet häufig mehr als eine Möglichkeit, um Strecken abzubilden. Diese Uneindeutigkeit erschwert das Identifizieren und Vergleichen von Streckenabschnitten.

Ein Beispiel sind die beiden Möglichkeiten, die Breite von Fahrstreifen zu beschreiben. Die erste Möglichkeit nutzt `<width>`-Elemente um die Fahrstreifenbreite direkt anzugeben. Die zweite Möglichkeit nutzt `<border>`-Elemente, um den Abstand der Außenseite des Fahrstreifens zur Referenzlinie anzugeben. Die Breite des Fahrstreifens ergibt sich dann aus der Differenz zum nächstinneren Fahrstreifen. Listings 2.1 und 2.2 enthalten zwei Minimalbeispiele, um den Unterschied zu verdeutlichen. Beide Varianten, die Fahrstreifenbreiten anzugeben, sind in der Formatspezifikation enthalten, so dass sie von jeder OpenDRIVE-fähigen Software unterstützt werden.

Listing 2.2: Fahrstreifenbreite, angegeben mittels `<border>`-Element.

```
<lane id="-1" type="driving" level="false">
  <border sOffset="0.0" a="3.0" b="0.0" c="0.0" d="0.0"/>
</lane>

<lane id="-2" type="shoulder" level="false">
  <border sOffset="0.0" a="5.0" b="0.0" c="0.0" d="0.0"/>
</lane>
```

Anders sieht es bei mit der Nutzung von `<userData>`-Elementen aus. Zwar ist explizit vorgesehen, dass Nutzer mit diesen Elementen ihre eigenen, selbst definierten, Strukturen umsetzen und damit eine große Flexibilität erhalten, allerdings führt ein solches Vorgehen schnell zu Kompatibilitätsproblemen. Eine dritte Software, welche OpenDRIVE nach Spezifikation interpretieren kann, wird individuell strukturier-

te <userData>-Elemente nicht oder nicht korrekt interpretieren können. Listings 2.3 und 2.4 enthalten zwei Minimalbeispiele, um den Unterschied zu verdeutlichen.

Listing 2.3: Regelbasierte Angabe der Fahrbahnmarkierungen.

```
<roadMark sOffset="0.0" type="broken" weight="standard"
          color="standard" width="0.12" laneChange="both"
          height="0.0">

  <type name="broken" width="0.12">
    <line length="3.0" space="6.0" tOffset="0.0"
          sOffset="0.0" rule="caution" width="0.12"/>
  </type>

</roadMark>
```

Listing 2.4: Explizite Angabe jeder einzelnen Fahrbahnmarkierung mittels <userData>-Element).

```
<roadMark sOffset="0.0" type="broken" weight="standard"
          color="standard" width="0.12" laneChange="both"
          height="0.0">

  <userData code="segmentData">
    <start sOffset="0.0" />
    <end sOffset="3.0" />
    <start sOffset="9.0" />
    <end sOffset="12.0" />
    <start sOffset="18.0" />
    <end sOffset="20.0" />
  </userData>

</roadMark>
```

2.3.3 Ableitung aus Vermessungsdaten

Im Gegensatz zu künstlich generierten Strecken, deren Referenzlinie mathematisch-perfekt und idealtypisch mit Geraden, Spiral- und Kreissegmenten modelliert ist und deren Fahrbahnmarkierungen in der Regel regelbasiert definiert werden, weisen aus Vermessungsdaten abgeleitete OpenDRIVE-Repräsentationen realer Streckenabschnitte – sogenannte *Realdaten* – deutlich „manuellere“ Züge auf. Referenzlinien werden durch Polynomfunktionen dritten Grades definiert, Fahrstreifenbreiten und Kurvenradien sind keinesfalls stabil, sondern schwanken – wenn auch im Rahmen von Bau- und Messgenauigkeit. Einer DLR[11]-internen Leistungsbeschreibung der Vermessung süddeutscher Autobahnen zufolge werden üblicherweise relative Genauigkeiten von 5 bis 10 cm sowie eine absolute Genauigkeiten von 20 cm erwartet.

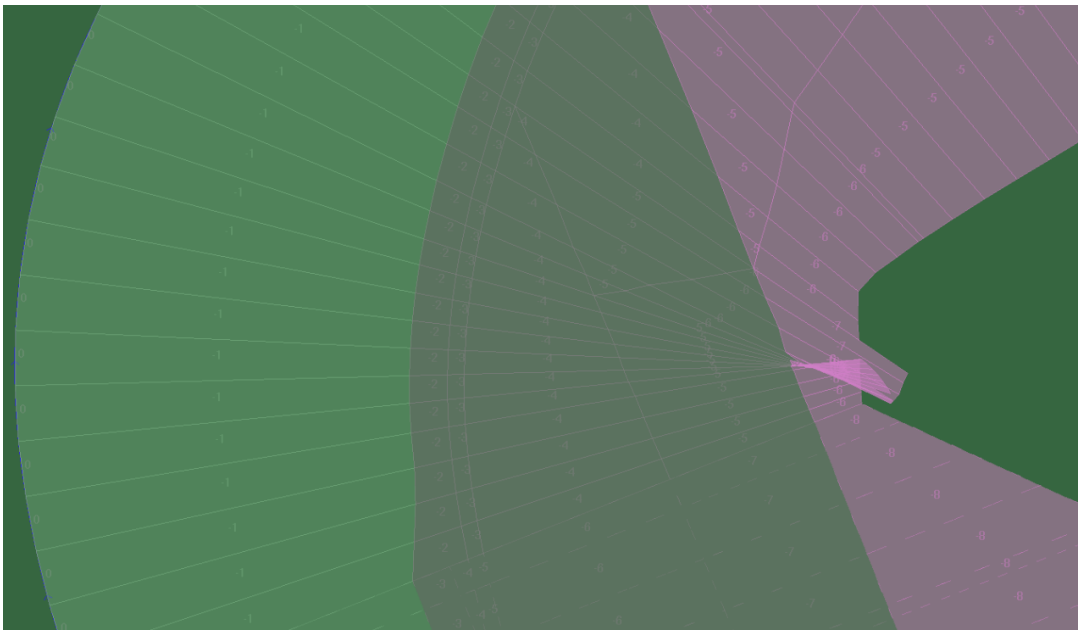
Ein weiteres Problem des OpenDRIVE-Formats tritt beim Modellieren von Strecken mit engen Kurven und schmalen Bereichen, etwa Bordsteinkanten, auf der Kurveninnenseite auf. In solchen Fällen kann es zu numerischen Problemen und geometrischen Artefakten kommen, wie in Abbildung 2.6 zu sehen ist. Diese Problematik kann zwar theoretisch auch bei künstlichen Szenen auftreten, lässt sich dort aber deutlich leichter durch eine Anpassung der Referenzlinie beheben als dies bei vermessungsbasierten OpenDRIVE-Daten der Fall ist.

2.3.4 Generierung künstlicher Strecken

Im Kontext szenarienbasierter Simulationen werden zahlreiche, auf Parameterbasis spezifizierte, OpenDRIVE-Dateien benötigt. Das manuelle Erstellen in Texteditoren oder kommerziellen OpenDRIVE-Editoren ist aufwendig und nicht parameterbasiert automatisierbar. Das liegt vor allem daran, dass Elemente sowohl relativ als auch absolut verknüpft sind und Änderungen an einer Stelle daher zahlreiche Änderungen an anderen Stellen nach sich ziehen würden.

Noyer et al. [18] haben die domänenspezifische Sprache *Simplified Road* definiert und eine zugehörige Programmbibliothek entwickelt, um Autobahnstreckenabschnitte

[11] Deutsches Zentrum für Luft- und Raumfahrt e.V.



(a) Schematische Darstellung. Zu sehen sind die Referenzlinie (lilafarbene Pfeile unter *lane zero*), ein Fahrstreifen (hellgrün, *lane -1* vom Typ `DRIVING`) sowie weitere *lanes* der Typen `BIKING`, `BORDER`, `SHOULDER` (grau, *lanes -2* bis *-7*) sowie `SIDEWALK` (pink, *lanes -4* bis *-8*)

(b) 3D-Rendering der OpenDRIVE-Szene.

Abbildung 2.6: Zwei Repräsentationen einer OpenDRIVE-Szene mit numerischem Artefakt auf der Innenseite eine engen Kurve.

im OpenDRIVE-Format stark verkürzt und vereinfacht darstellen zu können und die parameterbasierte, programmatische Generierung zu vereinfachen.

Für die Verkürzung werden Einschränkungen in Kauf genommen und Annahmen getroffen. Beispielsweise können Fahrstreifenanzahl und -breite nicht explizit angegeben werden und werden stattdessen aus der Angabe eines Regelquerschnitts (siehe Kapitel 2.1) abgeleitet. Die Beschreibung eines Streckenabschnitts mit Regelquerschnitt verkürzt sich dadurch ernorm, aber die Beschreibung eines Abschnitts mit nicht RAA-konformem Straßenquerschnitt ist nicht möglich.

Eine künstliche 5-km-Strecke, bestehend aus fünf `<road>`-Elementen (zwei Geraden, zwei Spiralbögen, ein Kreisbogen) kann in OpenDRIVE etwa 2000 Zeilen umfassen, wohingegen die entsprechende Simplified-Road-Datei mit lediglich gut 100 Zeilen auskommt. Die OpenDRIVE-Datei kann aus der Simplified-Road-Datei mittels der entsprechenden Simplified-Road-Java-Bibliothek generiert werden. Ausschnitte der entsprechenden Quelldateien sind in den Listings 2.5 und 2.6 zu sehen. Abbildung 2.7 zeigt ein 3D-Rendering der entsprechenden Szene.

Listing 2.5: Beispielhafte Simplified-Road-Notation eines 1-km-Streckenabschnitts (aus [18].)

```
1  <section length="1000">
2    <road crossSection="RQ36" mirror="true"/>
3    <course>
4      <arc radius="250"/>
5    </course>
6    <objects>
7      <infrastructure
8        type="noise barrier" t="0.5"
9        repeated="true" right="true"
10       center="false" left="true"/>
11   </objects>
12 </section>
```

Listing 2.6: Teile der aus Listing 2.5 generierten OpenDRIVE-Datei (aus [18].)

```
<road length="1000.0" id="0">
  <link>
    <successor elementType="road"
      elementId="1" contactPoint="start"/>
  </link>
  <type s="0.0" type="motorway"/>
  <planView>
    <geometry x="0.0" y="0.0" hdg="0.0"
      length="1000.0">
      <arc curvature="0.004"/>
  <elevationProfile>
  <lateralProfile/>
  <lanes>
    <laneSection>
      <left>
        <lane id="7" type="border">
        <lane id="6" type="shoulder">
        <lane id="5" type="stop">
        <lane id="4" type="driving">
          <link>
            <successor id="4"/>
          </link>
          <width a="3.75"/>
          <roadMark type="solid" weight="bold"
            color="white" width="0.3">
            <type>
              <line length="1.0" space="0.0"
                width="0.3"/>
            </type>
          </roadMark>
```

Abbildung 2.7: 3D-Rendering einer mit *Simplified Road* generierten OpenDRIVE-Beispielstrecke (aus [18]). Die Texturdaten sind nicht Teil der OpenDRIVE-Datei, sondern durch die Render-Software hinzugefügt worden.

Aus der Angabe `radius="250"` wird ein `<geometry>`-Element mit dem Attribut `curvature="0.004"` – ein Kurvenradius von $250\,\mathrm{m}$ entspricht einer Kurvenkrümmung von $0{,}004\,\mathrm{m}^{-1}$ (siehe Kapitel 2.1.2). Aus den Angaben `crossSection="RQ36"` wird die Fahrstreifen-Struktur des Regelquerschnitts 36 abgeleitet, welche in zahlreichen `<lane>`-Elementen resultiert, von denen nur einige in Listing 2.6 zu sehen sind.

Im Gegensatz zu OpenDRIVE-Dateien können Simplified-Road-Dateien einfach parametrisiert werden, indem statt konkreter Angaben Platzhalter (etwa `$RADIUS` statt 250) eingesetzt werden. Diese können dann mithilfe eines Skripts befüllt und die resultierenden Simplified-Road-Dateien einzeln an den Java-Generator weitergereicht werden. Die im ersten Absatz dieses Unterkapitels erwähnten notwendigen Anpassungen (relative und absolute Verknüpfung) entfallen so.

Die Arbeit von Noyer et al. befasst sich im Vergleich mit der vorliegenden Arbeit mit genau dem umgekehrten Vorgang. Statt in vorhanden Realdaten nach Abschnitten zu suchen, welche bestimmten Parameterräumen entsprechen, werden künstliche Abschnitte bestimmter Parameterausprägungen generiert. Dieser Punkt wird im Ausblick-Kapitel 5.2.5 wieder aufgegriffen.

2.3.5 GDAL-Erweiterung

Obwohl das in dieser Arbeit verwendete Dateiformat OpenDRIVE ein De-facto-Standard der Fahrsimulationsdomäne ist, mangelt es an der Unterstützung durch weitverbreitete Software zum Verarbeiten von Geodaten (etwa klassische Geoinformationssysteme). Einen Schritt in diese Richtung gehen Scholz und Idrobo [19] mit ihrer GDAL[12]-Erweiterung für OpenDRIVE. Die auf ihrer Arbeit basierende C++-Bibliothek [20] ermöglicht es, aus OpenDRIVE-Daten OGC[13]-konforme Geodaten zu erzeugen. So lässt sich beispielsweise aus einer OpenDRIVE-Datei die Referenzlinie extrahieren und in einem ESRI-Shapefile[14] speichern.

2.4 PEGASUS

Das Verbundprojekt PEGASUS entwickelt „Qualitätsstandards und Methoden für die Zulassung hochautomatisierter Fahrfunktionen“ [8]. Es beschäftigt sich mit der Frage, was ein automatisiertes Fahrzeug leisten muss und wie nachgewiesen werden kann, dass es dies zuverlässig tut [21]. Eines der Hauptziele von PEGASUS ist zudem die „Entwicklung einer durchgängigen und flexiblen Werkzeugkette zur Absicherung des automatisierten Fahrens“ [8].

[12] Geospatial Data Abstraction Library. Eine verbreitete Programmbibliothek zum Bearbeiten und Konvertieren von Geodaten.
[13] Open Geospatial Consortium. Gemeinnützige Organisation, die offene Standards für den Geodaten- und GIS-Bereich definiert.
[14] De-facto-Standard zur Speicherung von vektorbasierten Geodaten, ursprünglich von der Firma ESRI entwickelt.

2.4.1 Klassifizierung von Fahrfunktionen

Zur Klassifizierung des Automatisierungsgrads von Fahrfunktionen hat sich die sechsstufige Skala der Norm *SAE J3016* [22] etabliert. Sie enthält Werte von 0 bis 5. Dabei entspricht Stufe 0 *No Automation* und Stufe 5 *Full Automation*. Bis einschließlich Stufe 2 ist der Fahrer zur Umgebungsbeobachtung verpflichtet, bis einschließlich Stufe 3 muss der Fahrer zumindest als Rückfallebene bereitstehen. Eine detaillierte Übersicht bietet die Tabelle in Abbildung 2.8. Die Automatisierungsstufen in der Literatur auch als *automation levels* bezeichnet.

Im Rahmen von PEGASUS dient der sogenannte „Autobahn-Chauffeur“ als Beispiel einer Fahrassistenzfunktion. Er entspricht der SAE-Stufe 3 (*Conditional Automation*). Seine Funktionalität ist präzise definiert. Die Systemgrenzen umfassen eine niedrige Zahl an möglichen Manövern; etwa Staufolgefahrten, Notbremsungen und Ausweichverfahren, welche wiederum auf einzelne Elemente wie Ein- und Ausschervorgänge heruntergebrochen werden. Die Grundfunktionen und einige Systemgrenzen des Autobahn-Chauffeurs werden in Abbildung 2.9 dargestellt. Anhand dieser hypothetischen Assistenzfunktion entwickelt PEGASUS ein Verfahren zur Sicherstellung der Funktionsweise von Fahrfunktionen.

2.4.2 Die PEGASUS-Methode

Die Gesamtheit des im Rahmen von PEGASUS erarbeiteten Verfahren zum Testen von Fahrfunktionen wird als „PEGASUS-Methode“ bezeichnet. Ein wesentlicher Teil dieses Verfahrens ist Teil des Anwendungsfalls dieser Arbeit (siehe Kapitel 3.1) und wird daher im Folgenden näher ausgeführt: szenariobasierte Simulationen, wie Schuldt vorschlägt [6]. Der Aufbau der zugrundeliegenden Szenarien wurde bereits in Kapitel 2.2 beschrieben.

PEGASUS nutzt stochastische Methoden, um die Auswahl der Parameterausprägungen für konkrete Szenarien aus den Parameterräumen logischer Szenarien vorzunehmen [24, 25]. Um konkrete Szenarien durchzusimulieren, werden unter anderem OpenDRIVE-Dateien benötigt, welche die ersten drei Beschreibungsebenen (Straßenebene und Leitinfrastruktur, einschließlich vorübergehender Veränderungen wie Baustel-

Level	Name	Narrative definition	DDT		DDT fallback	ODD
			Sustained lateral and longitudinal vehicle motion control	OEDR		
Driver* performs part or all of the *DDT						
0	**No Driving Automation**	The performance by the *driver* of the entire *DDT*, even when enhanced by *active safety systems*.	*Driver*	*Driver*	*Driver*	n/a
1	**Driver Assistance**	The *sustained* and *ODD*-specific execution by a *driving automation system* of either the *lateral* or the *longitudinal vehicle motion control* subtask of the DDT (but not both simultaneously) with the expectation that the *driver* performs the remainder of the *DDT*.	*Driver* and *System*	*Driver*	*Driver*	Limited
2	**Partial Driving Automation**	The *sustained* and *ODD*-specific execution by a *driving automation system* of both the *lateral* and *longitudinal vehicle motion control* subtasks of the *DDT* with the expectation that the *driver* completes the *OEDR* subtask and *supervises* the *driving automation system*.	***System***	*Driver*	*Driver*	Limited
***ADS* ("*System*") performs the entire *DDT* (while engaged)**			*System*	***System***	*Fallback-ready user (becomes the driver during fallback)*	Limited
3	**Conditional Driving Automation**	The *sustained* and *ODD*-specific performance by an *ADS* of the entire DDT with the expectation that the *DDT fallback-ready user* is *receptive* to *ADS*-issued *requests to intervene*, as well as to *DDT performance-relevant system failures* in other *vehicle* systems, and will respond appropriately.				
4	**High Driving Automation**	The *sustained* and *ODD*-specific performance by an *ADS* of the entire *DDT* and *DDT fallback* without any expectation that a *user* will respond to a *request to intervene*.	*System*	*System*	***System***	Limited
5	**Full Driving Automation**	The *sustained* and unconditional (i.e., not *ODD*-specific) performance by an *ADS* of the entire *DDT* and *DDT fallback* without any expectation that a *user* will respond to a *request to intervene*.	*System*	*System*	*System*	**Unlimited**

Abbildung 2.8: Übersicht der Automatisierungsstufen nach *SAE J3016* (entnommen aus [22]).

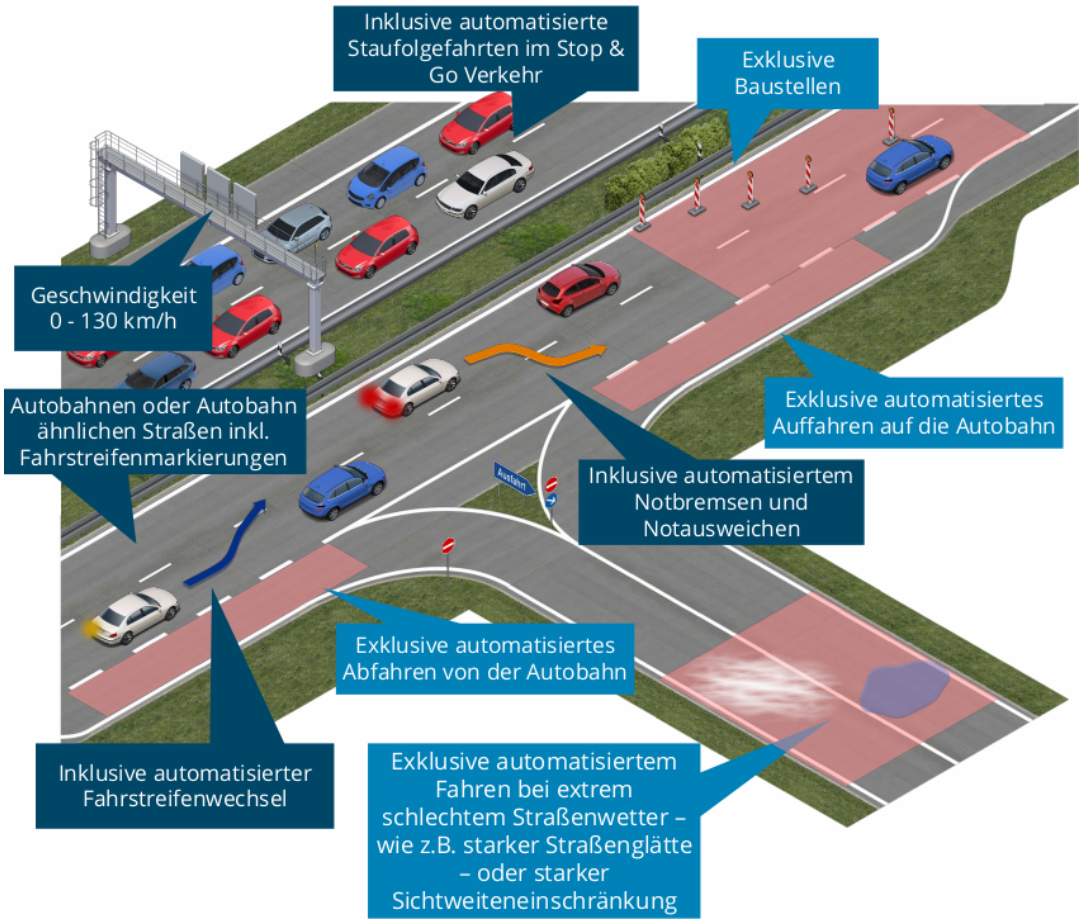


Abbildung 2.9: Grundfunktionen des Autobahn-Chauffeurs (aus [23]). Enthaltene Funktionen auf dunkelblauem, nicht enthaltene Funktionen auf hellblauem Hintergrund.

len) eines Szenarios abbilden. Diese Dateien werden unter Nutzung der in Kapitel 2.3.4 beschriebenen Arbeit von Noyer et al. [18, 26] generiert und zusammen mit den Parameterausprägungen der restlichen Beschreibungsebenen weitergereicht, um durchsimuliert zu werden.

An dieser Stelle greift diese Arbeit ein. Neben den mathematisch perfekten, künstlich generierten Streckenabschnitten sollen Simulationen in Realdaten-Streckenabschnitten, welche den Anforderungen logischer Szenarien entsprechen, durchgeführt werden können.

2.5 Verfahren zum Vergleichen von Linien-/Netzwerkstrukturen

Für die Entwicklung eines Identifikationsverfahrens für Streckenabschnitte mit bestimmten geometrischen Eigenschaften bietet sich die Konsultation von Arbeiten zum Vergleich von Straßennetzen und anderen linienartigen Strukturen an. Exemplarisch werden zwei Arbeiten über den Vergleich von Trajektorien beziehungsweise Straßennetzen vorgestellt.

In einer Studie vergleichen Wang et al. [27] Methoden zum Vergleich von Trajektorien. Trajektorien sind aufgezeichnete Bewegungsabläufe, im Wesentlichen also mit Zeitstempeln angereicherte Punktfolgen, die sich als Linienstruktur interpretieren lassen. Es stellt sich heraus, dass einige der vorgestellten Methoden, die weniger Gewicht auf die Zeitkomponente legen, beim Vergleich von realen Streckenabschnitten miteinander hilfreich sein könnten (siehe Ausblick-Kapitel 5.2.3). Die vorliegende Arbeit basiert allerdings auf logischen Szenarien (siehe Kapitel 2.2).

Beim Kartenabgleich (*map matching*) werden zwei Repräsentationen desselben Straßennetzes verglichen, Knoten und Kanten ihren Entsprechungen zuzuordnen. Mögliche Anwendungsfälle sind das Abbilden von Fahrzeug-Trajektorien auf die Straßennetz-Repräsentation eines Navigationssystems oder die Integration von Daten aus unterschiedlichen Quellen. Sämann beschäftigt sich in seiner Masterarbeit [28] mit der „Bestimmung einer Bewertungsmetrik zum Vergleich digitaler Straßennetze“. Dabei unter-

sucht er Edge-Matching-Verfahren, also Verfahren zum Kantenabgleich. Neben geometrischen Eigenschaften werden Attributdaten berücksichtigt. Seine Methoden fokussieren sich mehr auf die Topologie von Knoten und Kanten der Straßen*netze* statt auf die tatsächlichen geometrischen Ausprägungen wie Länge und Kurvenradien. Da sich auch die Anwendungsfelder – städtische Straßennetze statt isolierter Autobahnabschnitte – stark unterscheiden, könnte seine Arbeit erst bei einer möglichen zukünftigen Ausweitung über die Autobahnabschnitte hinaus hilfreich werden (siehe Ausblick-Kapitel 5.2.2).

2.6 Verwendete Programmbibliotheken

Der in dieser Arbeit entwickelte Algorithmus ist in der Programmiersprache Java[15] (JDK 1.8) implementiert worden. Neben nativen Standardbibliotheken wurden die *GeoTools*-Bibliothek[16] (Version 20.1) und die *JTS Topology Suite*[17] (Version 1.16) genutzt. Beides sind Programmbibliotheken zur Verarbeitung von Geodaten, insbesondere OGC-konformen Vektordaten. Außerdem fand eine DLR-interne Java-Implementierung der Arbeit von Scholz und Idrobo [19] (siehe Kapitel 2.3.5) Verwendung.

[15] https://www.java.com
[16] https://geotools.org
[17] https://github.com/locationtech/jts

3 Methodik

Dieses Kapitel beschreibt den Anwendungsfall, anhand dessen die eingangs gestellte Forschungsfrage untersucht und für den eine Suchfunktion für Streckenabschnitte implementiert wurde. Anschließend wird die Funktionsweise dieser Suchfunktion dargelegt und erklärt, wie die im nächsten Kapitel beschriebenen Ergebnisse entstanden sind und ausgewertet wurden.

3.1 Anwendungsfall und Datengrundlage

Der Anwendungsfall basiert auf Realdaten des *Digitalen Testfelds Autobahn*[1]. Enthalten sind zwei Abschnitte der deutschen Bundesautobahn 9 bei Ingolstadt, welche insgesamt etwa 117,671 km umfassen und ein Teil der Datengrundlage von PEGASUS sind. Diese Zahl berücksichtigt beide Fahrtrichtungen und Teile von Anschlussstellen. Eine Übersicht bietet Abbildung 3.1 und ein Beispiel für enthaltene Teile von Anschlussstellen findet sich in Abbildung 3.2.

Die Streckendaten liegen im OpenDRIVE-1.4-Format vor, welches in Kapitel 2.3 ausführlich behandelt wurde. Das in den Daten genutzte absolute (x, y)-Koordinatensystem basiert auf Meterangaben und ist – zwangsläufig – zweidimensional-kartesisch. Daher sind Längenangaben einfach zu interpretieren und Entfernungsberechnungen leicht durchzuführen.

Der OpenDRIVE-Datensatz unterteilt den Autobahnverlauf in möglichst lange Straßenelemente (`<road>`-Elemente). Eine Unterteilung findet lediglich an den Anschlussstellen statt. Ein langes Straßenelement verläuft also von einer Auffahrt bis zur nächs-

[1] https://mcloud.de/web/guest/suche/-/results/detail/E74607F2-1F17-4574-B895-8F3319A24E81

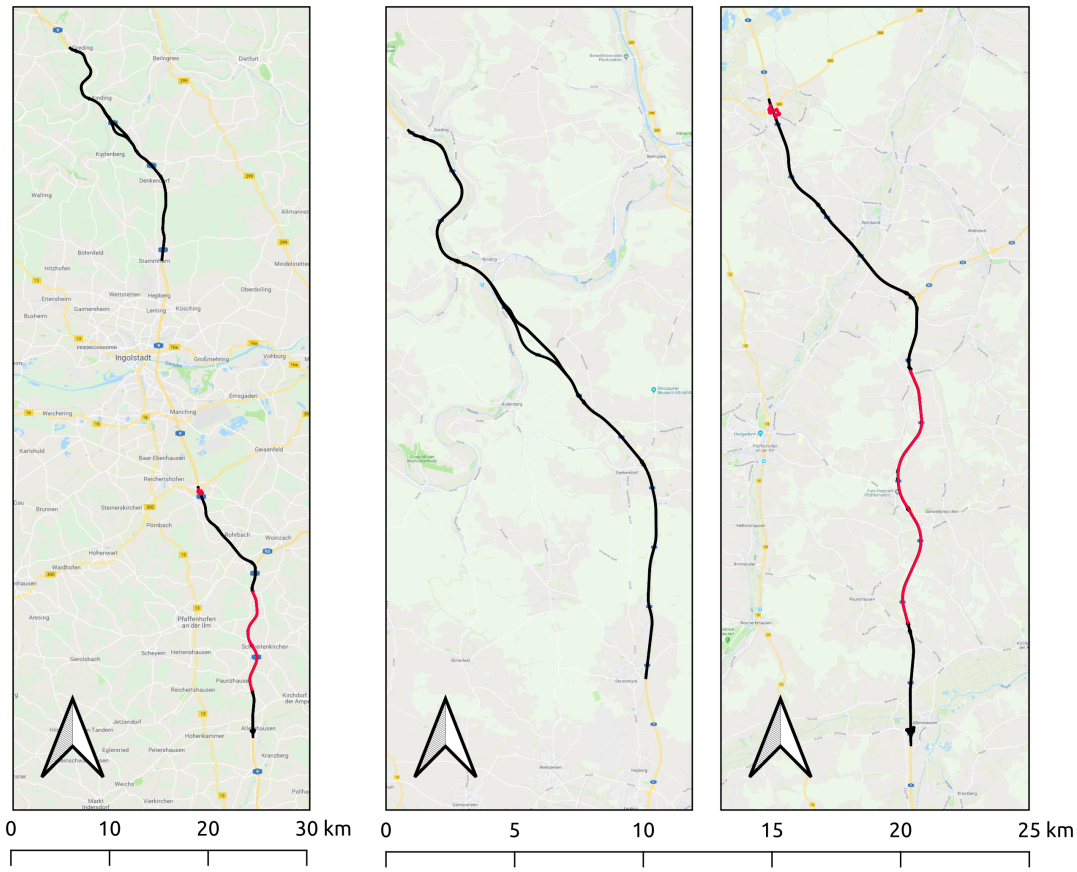


Abbildung 3.1: Links eine Übersichtskarte der beiden im untersuchten OpenDRIVE-Datensatz enthaltenen Streckenabschnitte der A 9 (schwarze Linien) bei Ingolstadt. Daneben ein detaillierterer Blick auf die Abschnitte nördlich (mittig) und südlich von Ingolstadt (rechts). Die roten Linien markieren die in den Abbildungen 3.2 und 4.6 gezeigten Ausschnitte. Hintergrund: Google Maps.

Abbildung 3.2: Referenzlinien der A 9 bei der Anschlussstelle Langenbruck. Zu sehen ist, dass neben Streckenabschnitte der Autobahn (schwarz) auch Teile der Auf- und Abfahrten (rot) enthalten sind.

ten Ausfahrt, woraufhin mehrere kleine Elemente zwischen Abfahrt und zugehöriger Auffahrt folgen, bis das nächste große Straßenelement anschließt. Da PEGASUS sich mit dem Autobahn-Chauffeur auf Streckenabschnitte ohne Auf- und Abfahrten beschränkt, genügt es, innerhalb einzelner großer Straßenelemente nach passenden Streckenabschnitten zu suchen. Um zukünftig nach Streckenabschnitten für komplexere Szenarien zu suchen, müsste `<road>`-Element-übergreifend verarbeitet werden. Bei der Spezifikation der Output-Schnittstelle wird diese Möglichkeit bereits berücksichtigt.

Zum Anwendungsfall gehört außerdem das Projekt PEGASUS, in dessen Werkzeugkette die entwickelte Funktionalität integrierbar sein soll. In Unterkapitel 3.2.3 finden sich daher Schnittstellendefinitionen.

3.1.1 Untersuchte Parameter

Aus den vorliegenden Realdaten lassen sich theoretisch zu vielen denkbaren Parametern der ersten beiden Szenario-Beschreibungsebenen (Straßeneigenschaften, Leitinfrastruktur) Informationen ableiten. Bei einigen, wie den Fahrbahnmarkierungen, führt die Flexibilität beziehungsweise Uneindeutigkeit von OpenDRIVE zu Problemen (siehe Kapitel 2.3.2). Tatsächlich spezifizierte logische PEGASUS-Szenarien konzentrieren sich bisher auf die Variation von Parametern der vierten Beschreibungsebene (Verkehrssituation und -dynamik). Die ersten beiden Beschreibungsebenen betreffend werden lediglich Längenangaben, die Straßenorientierung (Links-, Rechtskurve, Gerade) und teilweise ein Regelquerschnitt vorgegeben, weshalb diese Arbeit die Fragestellung anhand dieser Parameter untersucht. Da diese Parameter alle der ersten Beschreibungsebene zugeordnet sind, wird mit dem Vorhandensein von Leitplanken zusätzlich ein Parameter der zweiten Beschreibungsebene untersucht. Leitplanken sind relevant für die Funktionalität einiger On-Board-Sensoren von Fahrzeugen. Eine Ausweitung auf weitere Parameter und andere Datenquellen wird in Kapitel 5.2.2 besprochen.

Die entwickelte Software kann Streckenabschnitte anhand der Parameter *Mindestlänge*, *Straßenorientierung*, *Querschnitt*, *maximaler Kurvenradius* (entspricht der Angabe einer Mindestkrümmung) und *Leitplanken-Vorhandensein* identifizieren. Die Angabe von konkreten Werten für Mindestlänge und maximalen Kurvenradius scheint dem

wertebereichbasierten Konzept logischer Szenarien zu widersprechen. Diese Angaben bestimmen jedoch lediglich die unteren Grenzen von Wertebereichen. Dem Kriterium eines maximalen Kurvenradius von 1 500 m entsprechen alle Kurven mit einem Kurvenradius im Bereich 0–1 500 m. Der Vorgabe einer Mindestlänge von 2 km entsprechen alle ansonsten passenden Streckenabschnitte, die 2 km oder länger sind.

Aus einem erfolgreich identifizierten Streckenabschnitt der Länge 3 km könnten unendlich viele Teilabschnitte herausgeschnitten werden, die der Mindestlänge von 2 km genügen. Die Auswahl der Teilabschnitte wird dem Nutzer beziehungsweise der nächsten Stelle der Verwertungskette überlassen.

3.2 Umsetzung

In diesem Unterkapitel wird der entwickelte Algorithmus erklärt. Abbildung 3.3 stellt den groben Ablauf, der in den folgenden Absätzen und Unterkapiteln im Detail beschrieben wird, dar.

Die implementierte Suchfunktion nimmt eine reguläre OpenDRIVE-Datei und ein logisches Szenario entgegen. Die OpenDRIVE-Datei kann durch Nutzung des OpenDRIVE-XML-Schemas und der nativen Java-Bibliothek JAXB[2] in eine Java-Klassenstruktur geladen werden. Das logische Szenario wird aus einer Konfigurationsdatei importiert, auf deren Struktur Kapitel 3.2.3 eingeht.

Als erstes wird für jedes Straßenelement (`<road>`) eine Liste mit *Events* generiert. Ein Event markiert einen Bereich des Straßenelements, in dem bestimmte Parameterausprägungen, etwa der Kurvenradius, als konstant angenommen werden. Die vom Event enthaltenen Informationen und die Ableitung aus OpenDRIVE-Daten unterscheiden sich je nach betroffenem Parameter geringfügig und werden in Kapitel 3.2.1 im Detail beschrieben.

Sobald zusätzlich ein Szenario geladen wurde, durchläuft der Algorithmus die Eventliste jedes Straßenelements, um den Szenariovorgaben entsprechende Streckenabschnitte zu identifizieren und auf die Ergebnisliste zu setzen. Kapitel 3.2.2 beschreibt diesen

[2] Java Architecture for XML Binding.

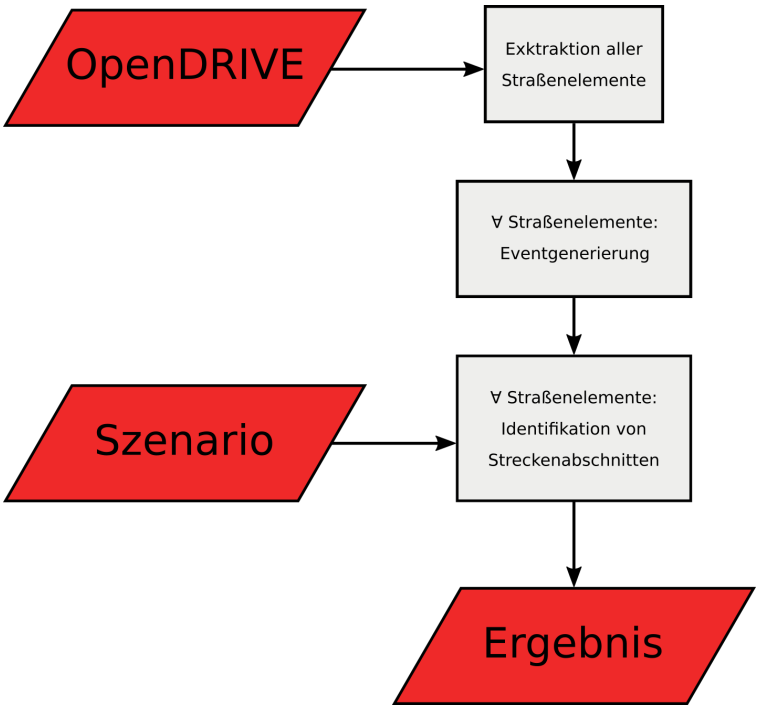


Abbildung 3.3: Grundlegender Ablauf der Identifikation von Streckenabschnitten. Die Eventgenerierung wird in Kapitel 3.2.1, der Szenario-Import in Kapitel 3.2.3, der Abgleich von Szenario und Events (die eigentliche Identifikation passender Streckenabschnitte) in Kapitel 3.2.2 und das Ausgabeformat in Kapitel 3.2.3 beschrieben.

Vorgang näher. Anschließend werden die Resultate zurückgegeben und auf strukturierte Art und Weise abgespeichert (siehe Kapitel 3.2.3).

3.2.1 Eventgenerierung

Es sind Events für die zu untersuchenden Parameter *Orientierung*, *maximaler Kurvenradius*, *Querschnitt* und *Leitplanken* implementiert. Für den Parameter *Mindestlänge* werden keine Events benötigt, da dieser erst zum Tragen kommt, wenn alle anderen Parameter den Szenariovorgaben entsprechen und daher eine Sonderrolle einnimmt (siehe Kapitel 3.2.2).

Ein Event besteht aus mehreren Attributen, von denen die Angabe des betroffenen Parameters das maßgebliche ist. Folglich kann zwischen Orientierungs-, Kurvenradius-, Querschnitts- und Leitplanken-Events unterschieden werden. Weitere Attribute sind der Start- und Endpunkt entlang der Referenzlinie eines Straßenelements, welche die räumliche Ausdehnung oder Gültigkeit des Events angeben. Hinzu kommen parameterabhängige Attribute, auf die in den kommenden Unterkapiteln eingegangen wird.

Sind alle Events für alle Parameter generiert, werden sie in Event-Listen (eine je Straßenelement) zusammengeführt und aufsteigend nach ihrem Startpunkt sortiert. Sie liegen nun „in Fahrtrichtung“ vor. Eventlisten sind die interne Grundlage des eigentlichen Identifikationsprozesses. Sie fallen als Zwischenprodukt an, welches zusätzlich als Ausgangspunkt für eine Parametrisierung von aus Vermessungsdaten abgeleiteten OpenDRIVE-Daten dienen kann. Mehr hierzu im Ausblick-Kapitel 5.2.5.

Querschnitt

OpenDRIVE beschreibt den Straßenquerschnitt in sogenannten *Querschnitts-Elementen* (`<laneSection>`). Laut Spezifikation [14] ist die Anzahl an *Fahrstreifen-Elementen* (`<lane>`) je Querschnittselement konstant. Lediglich ihre Breite kann variieren. Kommt ein neuer Fahrstreifen hinzu, beginnt sein `<lane>`-Element mit einer Breite von 0 m und verbreitert sich gemäß der angegebenen Polynomfunktion (siehe Kapitel 2.3.1). Das Verschwinden eines Fahrstreifens funktioniert analog.

Um die Anzahl der durchgehend vorhandenen Fahrstreifen abzuleiten wird die Breites jedes Fahrstreifens zu Beginn und am Ende des Querschnittselements berechnet. Befinden sich beide Werte nahe beieinander (und keiner bei $0\,\mathrm{m}$) wird ein Fahrstreifen als durchgehend betrachtet.

Je `<laneSection>`-Element wird ein Querschnittsevent erstellt, welches die Anzahl durchgehend vorhandener Fahrstreifen angibt. Ändert sich die Anzahl der Fahrstreifen, etwa weil sich das dritte Fahrstreifenelement im Verlauf des Querschnittselements von $0\,\mathrm{m}$ auf $3{,}75\,\mathrm{m}$ verbreitert, wird die Anzahl der durchgehenden Fahrstreifen um $0{,}5$ erhöht. Ein solcher Streckenabschnitt kann per Definition keinem Regelquerschnitt entsprechen. Als Start- und Endpunkt des Querschnittsevents werden Start- und Endpunkt des Querschnittselements angegeben.

Leitplanken

In den Daten des Anwendungsfalls werden Leitplanken als *Objekt-Elemente* (`<object>`) mit dem Attribut `name="guardRail"` repräsentiert, welche unabhängig von Geometrieelementen (`<geometry>`, Basis für die Ableitung von Orientierung und Kurvenradius) und Querschnittselementen (`<laneSection>`, Basis für die Ableitung des Querschnitts) sind. Leitplanken-Objektelemente im genutzten Datensatz können Instanzen (`<repeat>`) unterschiedlicher Länge enthalten, die nicht zwingend lückenlos aufeinander folgen.

Für jede Instanz, also jedes `<repeat>`-Element, wird ein Leitplankenevent erstellt, welches die Angabe „Leitplanken vorhanden“ enthält. Als Start- und Endpunkt werden Start- und Endpunkt der Instanz angegeben. Nachdem aus allen Leitplanken-Instanzen Leitplankenevents generiert wurden, werden für gegebenenfalls vorhandene Lücken in der Abdeckung des Straßenelementverlaufs weitere Leitplankenevents generiert, welche die Angabe „keine Leitplanken vorhanden“ enthalten. Um eigentlich durchgehende Leitplanken nicht durch minimale Lücken, die durch Rundungsfehler oder Ungenauigkeiten bei der Vermessung entstehen können, unterbrechen zu lassen, wird ein Grenzwert von $1\,\mathrm{m}$ angenommen. Erst, wenn eine Lücke diesen überschreitet, wird sie als solche gewertet und ein entsprechend „negatives“ Leitplankenevent erstellt.

Nicht berücksichtigt wird, ob Leitplanken auf beiden Seiten der Fahrbahn vorhanden sind. Dieses könnte zukünftig aus ihrer relativen Positionen zur Referenzlinie in der t-Dimension abgeleitet werden.

Orientierung und Kurvenradius

Die Ableitung der Straßenorientierung und des Kurvenradius geschieht auf Basis der Geometrieelemente (`<geometry>`). Je Geometrieelement werden zwei Events erstellt: ein Orientierungs- und ein Kurvenradiusevent. Beiden wird die Ausdehnung des entsprechenden Geometrieelements übertragen und entweder eine Kurvenradiusangabe oder eine Orientierungsangabe angefügt. Um diese Eigenschaften des Geometrieelements einfacher bestimmmen zu können, wird es aus der mathematisch kontinuierlichen Darstellung in einen diskreten Linienzug (*LineString*) umgewandelt (siehe Kapitel 2.6). Dieser basiert auf dem meterbasierten, kartesischen (x, y)-Koordinatensystem des verwendeten OpenDRIVE-Datensatzes und ermöglicht daher die Anwendung einfacher geometrischer Berechnungen.

Der Kurvenradius eines Geometrieelements wird bestimmt, indem ein Kreis durch Start-, Mittel- und Endpunkt des Linienzugs konstruiert und dessen Radius bestimmt wird. Verfeinerte Methoden der Ableitung des Kurvenradius, etwa feiner auflösend und glättend, werden im Ausblick (Kapitel 5.1) besprochen. Die Orientierungsangabe kann drei Ausprägungen annehmen: *Linkskurve*, *Gerade* und *Rechtskurve*. Es wird angenommen, dass die Ausprägung innerhalb eines Geometrieelements konstant ist. Bestimmt wird sie durch die unterschiedliche Steigung der Liniensegmente zwischen Start- und Mittel- beziehungsweise Mittel- und Endpunkt des Linienzuges. Sind die Steigungen der beiden Liniensegmente identisch, handelt es sich um eine Gerade, ist die Steigung des zweiten Liniensegments größer als die des ersten, handelt es sich um einen Linksknick und anderenfalls um einen Rechtsknick.

Formel 3.1 zeigt den mathematischen Ausdruck, aus dessen Vorzeichen die Orientierung abgeleitet wird. (x_1, y_1) sind dabei die Koordinaten des Startpunkts, (x_2, y_2) die

des Mittelpunkts und (x_3, y_3) die des Endpunkts des betrachteten Linienzugs.

$$(y_2 - y_1) \cdot (x_3 - x_2) - (y_3 - y_2) \cdot (x_2 - x_1) \tag{3.1}$$

Keines der Geometrieelemente des Datensatzes weist einen unendlichen Kurvenradius auf. Das bedeutet, dass sich im genutzten Datensatz erwartungsgemäß (siehe Kapitel 2.3.3) kein einziger perfekt gerader Abschnitt befindet. In dieser Arbeit werden daher Geometrieelemente mit einem Kurvenradius über $10\,000\,\mathrm{m}$, also einer geringeren Krümmung als $0{,}000\,1\,\mathrm{m}^{-1}$ als Gerade gewertet. In einer zukünftigen Umsetzung könnte dieser Grenzwert als ein zur Prozessierung extern konfigurierbarer implementiert werden.

3.2.2 Identifikation von Streckenabschnitten

Ausgangspunkt der Identifikation von passenden Streckenabschnitten sind ein Szenario mit seinen Parametervorgaben und die vorsortierte Eventliste eines Straßenelements.

Der Algorithmus arbeitet die komplette Eventliste des untersuchten Straßenelements „in Fahrtrichtung“, also entlang seiner Referenzlinie durch. Bei jedem Event hält er an, prüft, ob es die Szenariovorgaben erfüllt oder nicht und aktualisiert die entsprechende Statusvariable. Zeigen zu einem Zeitpunkt alle für das Szenario relevanten Statusvariablen an, dass die Vorgaben erfüllt sind, merkt sich der Algorithmus die aktuelle s-Koordinate und fährt fort. Sobald sich dieser Zustand durch ein den Szenariovorgaben widersprechendes Event ändert, wird geprüft, ob die seit der zuletzt gemerkten s-Koordinate zurückgelegte Strecke eine gegebenenfalls vorgegebene Mindestlänge des Szenarios erreicht oder übersteigt. In diesem Fall wird ein Eintrag in der Ergebnisliste vorgenommen. In jedem Fall werden die Statusvariablen aktualisiert und fortgefahren, bis wieder alle Szenariovorgaben erfüllt sind oder das Ende des Straßenelements erreicht ist.

Abgleich der Parameter im Einzelnen

Die A 9 ist eine der ältesten überregionalen Autobahnen Deutschlands. Obwohl sie regelmäßig ausgebaut und modernisiert wurde, können nicht alle Abschnitte den aktuell gültigen Regelquerschnitten der Richtlinien für die Anlage von Autobahnen (RAA, siehe Kapitel 2.1) entsprechen. Einige Teile werden daher entweder den Regelquerschnitten der Vorgänger-Richtlinien oder gar keinem Regelquerschnitt exakt entsprechen. Da PEGASUS-Szenarien RAA-basiert sind und so lediglich aktuell gültige Regelquerschnitte umfassen, kann es hier zu Problemen kommen. Um das wichtige Kriterium des Straßenquerschnitts überhaupt sinnvoll anwenden zu können, leitet die implementierte Suchfunktion den Regelquerschnitt aus seiner maßgeblichen Eigenschaft, der Anzahl Fahrstreifen, ab. Auf einen Abgleich der Breiten von Fahr-, Standstreifen und des Banketts wird verzichtet. Dies ist ein einfacher Ansatzpunkt, um die Suche zu verfeinern (siehe Ausblick-Kapitel 5.2).

Ob ein Event den Leitplanken-Vorgaben eines Szenarios entspricht wird durch einen Vergleich der entsprechenden Angaben, *ja*/`1` oder *nein*/`0`, festgestellt. Genauso einfach wird bei der Orientierung verfahren. Beim Vergleich eines Kurvenradiusevents mit dem gegebenen Szenario wird geprüft, ob der errechnete Kurvenradius den Kurvenradiusgrenzwert des Szenarios übersteigt oder nicht.

3.2.3 Schnittstellen-Definition

Die entwickelte Software benötigt Streckendaten im OpenDRIVE-Format und ein logisches Szenario, um daraus eine Liste mit identifizierten Streckenabschnitten zurückzugeben. Eingeladen werden können reguläre OpenDRIVE-1.4-Dateien, welche üblicherweise die Dateieindung `*.xodr` tragen.

PEGASUS strebt die Repräsentation von Szenarien durch das Dateiformat OpenSCENARIO an. Mangels fertiger PEGASUS-interner Schnittstellendefinitionen, verwendet die im Rahmen dieser Arbeit entwickelte Software einfache Konfigurationsdateien für die Repräsentation logischer Szenarien. Deren Spezifikation wird im nächsten Unterkapitel ausgeführt.

Die identifizierten Streckenabschnitte werden als `*.csv`-Datei zurückgegeben, um

Tabelle 3.1: Übersicht möglicher Schlüssel und Werte in Szenario-Konfigurationsdateien.

Schlüssel	mögliche Werte	Szenario-Parameter
`CROSSSECTION`	`RQ31`, `RQ36` oder `RQ43,5`	Regelquerschnitt
`GUARDRAILS`	`0` oder `1`	Leitplanken
`LENGTH_MIN`	Fließkommazahl	Mindestlänge (in Metern)
`ORIENTATION`	`LEFT`, `STRAIGHT` oder `RIGHT`	Orientierung
`RADIUS_MAX`	Fließkommazahl	maximaler Kurvenradius (in Metern)

möglichst universell weiterverwendbar zu sein. Der Inhalt dieser Dateien wird in Kapitel 3.2.3 beschrieben.

Inputspezifikation für Szenarien

Den Konfigurationsdateien, mit denen logische Szenarien beschrieben werden, liegt eine einfach Syntax zugrunde. Sie bestehen aus einer Reihe von Schlüsselwertpaaren. Je Zeile kann die Konfigurationsdatei eines dieser Paare enthalten. Schlüssel und Wert werden durch ein Gleichheitszeichen (=) voneinander getrennt. Der Schlüssel steht links, der Wert rechts des Gleichheitszeichens. Leerzeilen und Zeilen, die mit einem Doppelkreuz (#) beginnen werden beim Einlesen ignoriert. Die Dateiendung ist nicht relevant, aber in den Beispielen dieser Arbeit wird `*.properties` verwendet.

Tabelle 3.1 listet auf, welche Schlüssel welchen Szenarioparametern entsprechen und welche Werte sie annehmen können. Als Dezimalkennzeichen für Fließkommazahlen können sowohl Kommata als auch Punkte verwendet werden. Listing 3.1 zeigt die Konfigurationsdatei für ein Szenario welches Linkskurven mit einem maximalen Kurvenradius von 2 km, einer Mindestlänge von 400 m und mit einem dem Regelquerschnitt 36 entsprechenden Querschnitt beinhaltet, bei dem aber kein Wert darauf gelegt wird, ob Leitplanken vorhanden sind oder nicht.

Listing 3.1: Mögliche Inhalte einer Szenario-Konfigurationsdatei.

```
1 LENGTH_MIN=400
```

Tabelle 3.2: Übersicht der Einträge in einer CSV-Datei identifizierter Streckenabschnitte.

CSV-„Spalte“	Bedeutung
START_ROADID	id des <road>-Elements, auf dem der Streckenabschnitt beginnt
START_S	Position entlang der Referenzlinie, an der der Streckenabschnitt beginnt
END_ROADID	id des <road>-Elements, auf dem der Streckenabschnitt endet
END_S	Position entlang der Referenzlinie, an der der Streckenabschnitt endet
LENGTH	Länge des Streckenabschnitts
TIMESTAMP	Zeitstempel der Identifikation
XODR	Dateiname oder -pfad der zugrundeliegenden OpenDRIVE-Datei
SCNR	Dateiname oder -pfad der zugrundeliegenden Szenario-Datei

```
2 ORIENTATION=LEFT
3 RADIUS_MAX=2000
4 CROSSSECTION=RQ36
5 #GUARDRAILS=0
```

Outputformat

Da PEGASUS bisher keine Erweiterung um Realdaten-Streckenabschnitte vorsieht, ist keine diesbezügliche Schnittstelle spezifiziert. Daher werden die gefundenen Streckenabschnitte in einem möglichst einfachen Format ausgegeben: dem CSV-Format[3]. Es handelt sich um ein weitverbreitetes, aber simples Format zur strukturierten Datenspeicherung, vergleichbar mit einer Tabelle. Die erste Zeile einer CSV-Datei entspricht dem Tabellenkopf, jede weitere einer Tabellenzeile. Die Inhalte von Zellen werden durch Kommata oder Semikola voneinander getrennt. Tabelle 3.2 zeigt, unter Angabe welcher Informationen erfolgreich identifizierte Streckenabschnitte in einer CSV-Datei abgelegt werden. Listing 3.2 zeigt beispielhaft den Inhalt einer Ergebnisdatei.

Listing 3.2: Beispiel einer CSV-Datei mit identifizierten Streckenabschnitten, gerundeten Positions- und Längenangaben und ohne Zeitstempel.

[3] Comma-separated values.

```
START_ROADID;START_S;END_ROADID;END_S;LENGTH;TIMESTAMP;XODR;SCNR
2104000;0.0;2104000;587.3;587.3;;ex.xodr;scnrFile.properties;
2104000;1658.5;2104000;2723.6;1065.1;;ex.xodr;scnrFile.properties;
2104000;4206.9;2104000;4643.4;436.4;;ex.xodr;scnrFile.properties;
```

3.3 Ergebnisevaluation

Um den in diesem Kapitel vorgestellen Algorithmus und seine Implementierung zu evaluieren, werden in Kapitel 4 Auswertungsergebnisse präsentiert und anschließend in Kapitel 5 diskutiert. Für jedes der Szenarioparameter Querschnitt, Orientierung, Kurvenradius und Leitplanken wird abhängig von der Mindestlänge untersucht, wieviele Streckenabschnitte im Datensatz des Anwendungsfalls identifiziert werden und wieviel Streckenkilometer diese insgesamt abdecken. Die Plausibilität der Ergebnisse wird diskutiert und für den Parameter Kurvenradius exemplarisch eine grafische Überprüfung vorgenommen.

4 Ergebnisse

Bevor die in Kapitel 3.3 beschriebene Auswertung beginnt, werden einige grundlegende Statistiken über den der Analyse zugrundeliegenden OpenDRIVE-Datensatz vorgestellt. Anschließend werden die Ergebnisse je Parameter in Unterkapiteln angeführt und nach Möglichkeit in Relation zum kompletten Datensatz betrachtet.

Die Ergebnisse je Parameter werden größtenteils in Tabellen und darauf basierenden Diagrammen dargestellt. Die Anzahl gefundener Streckenabschnitte wird ausschließlich für den Parameter Kurvenradius näher untersucht, da die Ergebnisse einer Interpretation bedürfen. Zusätzlich wird für alle Parameter die *Gesamtstrecke* aller gefundenen Streckenabschnitte, abhängig von der Mindestlänge des Szenarios, untersucht. Für die Ermittlung dieser Gesamtstrecke werden die Längen aller als szenariokonform identifizierten Streckenabschnitte aufaddiert.

Die Mindestlängen betragen 0–10 km. Es ist zu erwarten, dass die Länge keines identifizierten Streckenabschnitts die des längsten Straßenelements (`<road>`), also 8,353 km (siehe Tabelle 4.1), übersteigt, da lediglich Teilstrecken von Straßenelementen identifiziert werden können. Außerdem müssen sowohl Anzahl als auch Gesamtstrecke identifizierter Streckenabschnitte mit steigender Mindestlänge monoton fallen.

Tabelle 4.1: Statistiken des A9-Testfelds und seiner beiden Teilbereiche.

	`<road>`-Elemente	längstes `<road>`-Element	Gesamtlänge
A9 Nord	85	8,353 km	57,497 km
A9 Süd	163	5,726 km	60,174 km
Gesamt	248	8,353 km	117,671 km

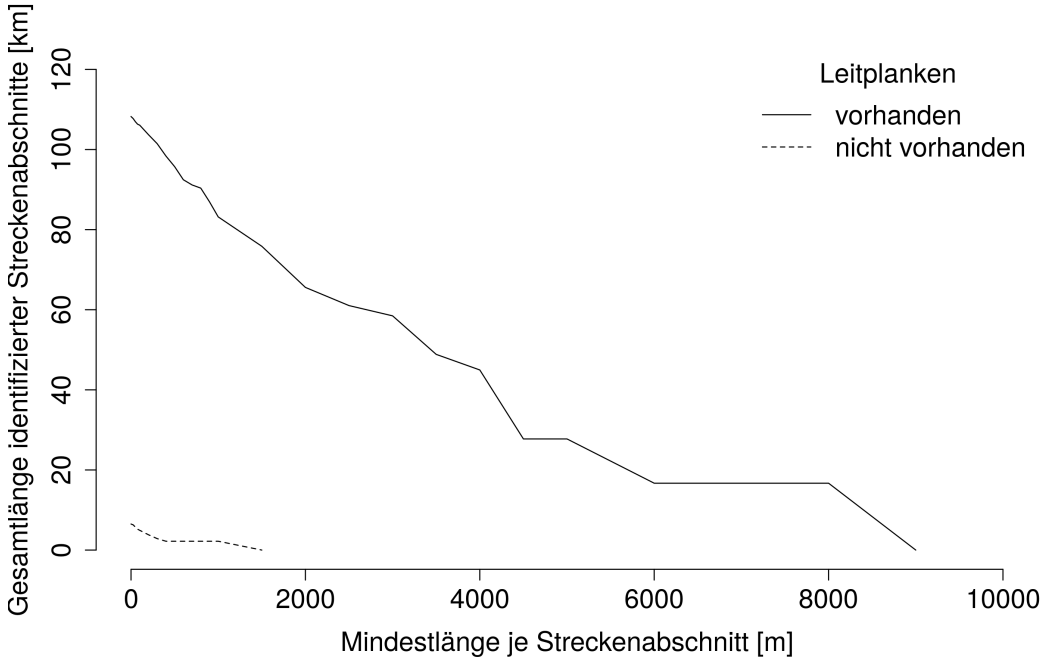


Abbildung 4.1: Gesamtlänge identifizierter Streckenabschnitte nach Mindestlänge und Leitplanken-Vorhandensein. Datengrundlage: Tabelle 4.2.

4.1 Leitplanken und Querschnitt

Abbildungen 4.1 und 4.2 zeigen die Gesamtlänge aller gefunden Streckenabschnitte nach Leitplanken-Vorhandensein beziehungsweise Regelquerschnitt und Mindestlänge. Sie basieren auf den Daten der Tabellen 4.2 und 4.3, welche die Ergebnisse einer Auswertung zahlreicher Szenarien mit exakt den angegebenen Parameterausprägungen sind.

In Abbildung 4.1 sind zwei Linien zu sehen, von denen die durchgezogene die Gesamtstrecke (zweite Achse) aller identifizierten Streckenabschnitte *mit* durchgehend vorhandenen Leitplanken, abhängig von der geforderten Mindestlänge (erste Achse), darstellt. Die gestrichelte Linie ist das Gegenstück für Streckenabschnitte *ohne* Leitplanken. Beide Linien verlaufen, wie erwartet, monoton fallend und weisen ihre letzten positiven Werte bei einer Mindestlänge von 8 km oder weniger auf.

Außerdem wird ersichtlich, dass ein Großteil der Gesamtstrecke mit Leitplanken versehen ist, was für eine Überregionalautobahn mit einer Richtgeschwindigkeit von 120 km/h nicht anders zu erwarten ist. Lediglich etwa 6,5 km des Datensatzes weisen keine Leitplanken auf, was auf die Straßenelemente der Auf- und Abfahrten zurückzuführen sein kann, die ebenfalls im Datensatz vorhanden sind (siehe Kapitel 3.1 und Abbildung 3.2).

Abbildung 4.2 weist drei Linien auf, von denen die durchgezogene die Gesamtstrecke der Ergebnisse für zweistreifige (Regelquerschnitt 31), die gestrichelte die Ergebnisse für dreistreifige (RQ 36) und die gepunktete die Ergebnisse für vierstreifige (Regelquerschnitt 43,5) Streckenabschnitte darstellt, jeweils abhängig von der geforderten Mindestlänge. Alle drei Linien verlaufen erwartungsgemäß monoton fallend und weisen ihre letzten positiven Werte bei einer Mindestlänge von 8 km oder weniger auf.

Der Abbildung und Tabelle 4.3 kann entnommen werden, dass der Großteil des Datensatzes eine dreistreifige Fahrbahn aufweist (etwa 94,2 km), aber vereinzelt auch zwei- (etwa 4,6 km) oder vierstreifig (etwa 8,7 km) verläuft. Die Summe dieser Gesamtstrecken beträgt etwa 107,5 km und damit rund 10,2 km weniger als die Gesamtstrecke des Datensatzes. Dies lässt sich einerseits durch die bereits erwähnten Straßenelemente der Anschlussstelleninfrastruktur erklären, welche typischerweise einstreifig verlaufen. Hinzu kommen all die Streckenabschnitte, während derer ein Fahrstreifen hinzukommt oder verschwindet, welche keiner Regelquerschnittvorgabe entsprechen können (siehe Kapitel 3.2.1).

4.2 Orientierung

Abbildung 4.3 weist drei Linien auf, von denen die durchgezogene die Gesamtstrecke der Ergebnisse für Geraden, die gestrichelte für Rechtskurven und die gepunktete für Linkskurven darstellt, jeweils abhängig von der geforderten Mindestlänge. Alle drei Linien verlaufen, wie erwartet, fallend. Ihre letzten positiven Werte weisen sie bei Mindestlängen von 1 km (Linkskurven) beziehungsweise 1,5 km (Rechtskurven, Geraden) auf.

Die Gesamtstrecken mit einer Mindeststrecken-Anforderung von 0 km summieren

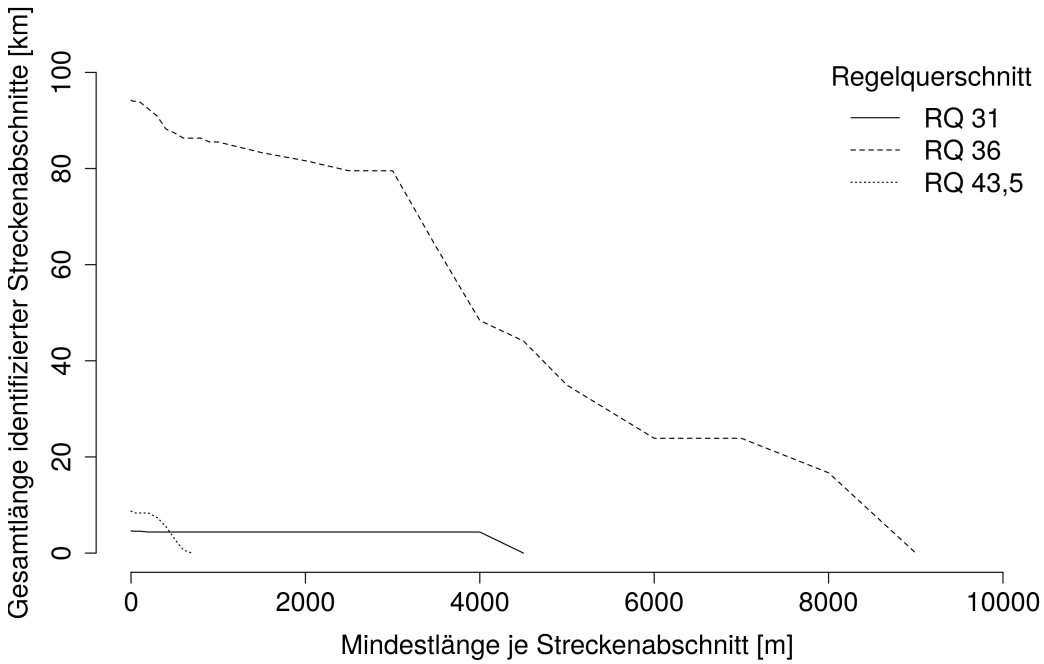


Abbildung 4.2: Gesamtlänge identifizierter Streckenabschnitte nach Mindestlänge und Querschnitt. Datengrundlage: Tabelle 4.2.

Tabelle 4.2: Anzahl (Spalten 2–3) und Gesamtstrecke (Spalten 4–5) identifizierter Streckenabschnitte nach Mindestlänge (d_{min}) und Leitplanken-Vorhandensein.

d_{min} [m]	mit Leitpl.	ohne Leitpl.	mit Leitpl. [m]	ohne Leitpl. [m]
0	198	68	108237	6502
25	117	43	107717	6290
50	97	28	106940	5732
75	87	19	106329	5208
100	84	16	106072	4928
200	68	8	103705	3822
300	59	4	101456	2858
400	50	2	98385	2206
500	44	2	95730	2206
600	38	2	92454	2206
700	36	2	91158	2206
800	35	2	90360	2206
900	31	2	86943	2206
1000	27	2	83174	2206
1500	21		75840	
2000	15		65574	
2500	13		61036	
3000	12		58472	
3500	9		48855	
4000	8		44963	
4500	4		27746	
5000	4		27746	
6000	2		16694	
7000	2		16694	
8000	2		16694	
9000				

Tabelle 4.3: Anzahl (Spalten 2–4) und Gesamtstrecke (Spalten 5–7) identifizierter Streckenabschnitte nach Mindestlänge (d_{min}) und Regelquerschnitt.

d_{min} [m]	RQ 31	RQ 36	RQ 43.5	RQ 31 [m]	RQ 36 [m]	RQ 43.5 [m]
0	9	82	54	4619	94189	8732
25	2	52	26	4550	94015	8506
50	2	52	20	4550	94015	8339
75	2	51	20	4550	93947	8339
100	2	50	20	4550	93855	8339
200	1	40	20	4379	92399	8339
300	1	34	16	4379	90917	7393
400	1	26	11	4379	88224	5540
500	1	24	5	4379	87388	2919
600	1	22	1	4379	86333	643
700	1	22		4379	86333	
800	1	22		4379	86333	
900	1	21		4379	85521	
1000	1	21		4379	85521	
1500	1	19		4379	83304	
2000	1	18		4379	81623	
2500	1	17		4379	79536	
3000	1	17		4379	79536	
3500	1	12		4379	63679	
4000	1	8		4379	48385	
4500		7			44134	
5000		5			34933	
6000		3			23882	
7000		3			23882	
8000		2			16699	
9000						

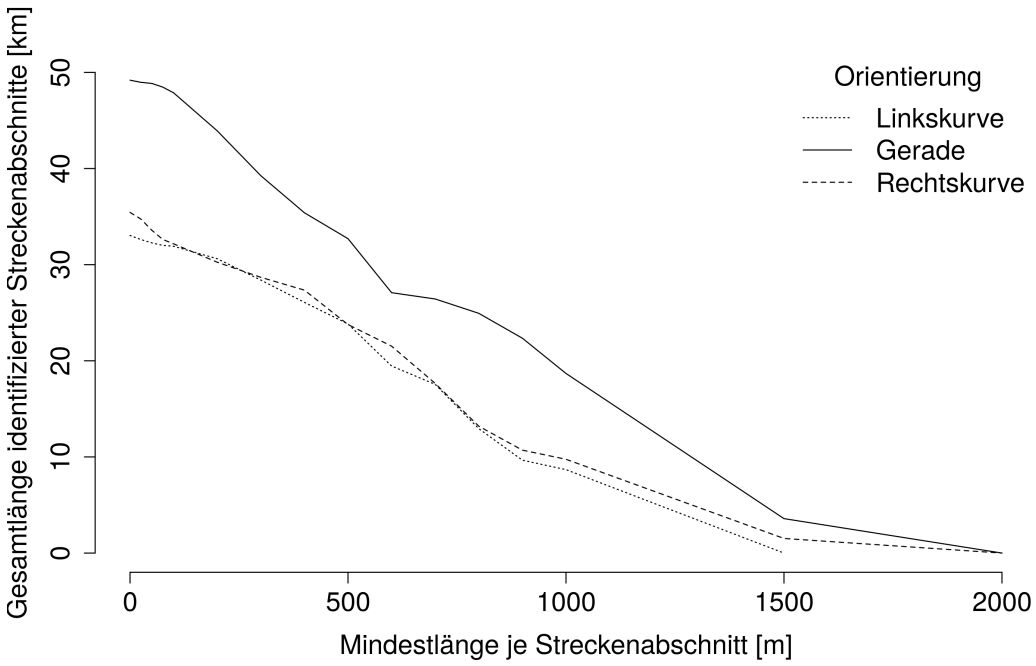


Abbildung 4.3: Gesamtlänge identifizierter Streckenabschnitte nach Mindestlänge und Kurvenrichtung. Datengrundlage: Tabelle 4.4.

Tabelle 4.4: Anzahl (Spalten 2–4) und Gesamtstrecke (Spalten 5–7) identifizierter Streckenabschnitte nach Mindestlänge (d_{min}) und Orientierung. Datengrundlage für Abbildung 4.3.

d_{min} [m]	Linkskurve	Gerade	Rechtskurve	Linksk. [m]	Gerade [m]	Rechtsk. [m]
0	175	175	228	33038	49193	35438
25	74	114	111	32614	48967	34740
50	64	111	81	32267	48853	33581
75	60	105	65	32012	48473	32626
100	59	98	60	31914	47874	32166
200	50	71	46	30620	43923	30255
300	41	52	40	28399	39256	28685
400	34	41	36	26079	35398	27352
500	29	35	28	23830	32703	23772
600	21	25	24	19457	27085	21524
700	18	24	18	17549	26430	17646
800	12	22	12	12942	24937	13196
900	8	19	9	9662	22348	10707
1000	7	15	8	8678	18684	9756
1500		2	1		3582	1519
2000						

sich auf 117,669 km, was bis auf eine Differenz von 0,002 km ziemlich genau genau der Gesamtstrecke aller Straßenelemente entspricht. Diese Differenz ist der Rundung der Zahlen auf ganze Meter zu verdanken.

Anteilsmäßig führen die geraden Abschnitte (41,8 %) gegenüber den Links- und Rechtskurven (28,1 % und 30,1 %), wobei die unterschiedlich orientierten Kurven zusammen 58,2 % der Gesamtstrecke ausmachen.

4.3 Kurvenradius

Die Ergebnisse der Kurvenradius-Auswertung werden in zwei Diagrammen (Abbildungen 4.4 und 4.5) sowie einem Satz vergleichender Karten (Abbildung 4.6) vorgestellt.

Insgesamt wird der Verlauf von Anzahl und Gesamtstrecke identifizierter Streckenabschnitte betrachtet sowie eine grafische Überprüfung der Kurvenradius der Streckenidentifikation durchgeführt.

In Abbildung 4.4 sind sechs Linien zu sehen, welche die *Anzahl* (zweite Achse) der identifizierten Streckenabschnitte abhängig vom maximalen Kurvenradius (unterschiedliche Linien) und der vom Szenario geforderten Mindestlänge (erste Achse) darstellen. Die dick gezeichnete, durchgezogene Linie repräsentiert *keine Beschränkung* des Kurvenradius. Die dünneren, unterschiedlich gemusterten Linien geben die Ergebnisse der maximalen Kurvenradien $10\,\mathrm{km}$, $2\,\mathrm{km}$, $1{,}5\,\mathrm{km}$, $1\,\mathrm{km}$ und $0{,}72\,\mathrm{km}$ wieder. Streckenabschnitte mit einem Kurvenradius über $10\,\mathrm{km}$ werden im Rahmen dieser Arbeit als Geraden interpretiert (siehe Kapitel 3.2.1) und $0{,}72\,\mathrm{km}$ ist der Mindestradius für Kurven von EKA-1-Autobahnen wie der A 9 gemäß der Richtlinien für die Anlage von Autobahnen (siehe Kapitel 2.1.2).

Die Linien sind, erwartungsgemäß, monoton fallend. Auch weisen sie ihre letzten positiven Datenpunkte bei einer Mindestlänge von $8\,\mathrm{km}$ auf, wie aus Tabellen 4.5 ersichtlich wird. Beides entspricht den Erwartungen.

Da eine Kurve mit einem Radius von beispielsweise $1{,}1\,\mathrm{km}$ sowohl die Bedingung, einen Kurvenradius von $1{,}5\,\mathrm{km}$ nicht zu überschreiten, als auch die Bedingung, einen Kurvenradius von $2\,\mathrm{km}$ nicht zu überschreiten, erfüllt, entspricht es der Erwartung, dass sich die einzelnen Linien nicht kreuzen. Die dick gedruckte Linie, welche das Ergebnis ohne beziehungsweise mit unendlichem Kurvenradiusmaximum repräsentiert, verläuft jedoch nicht abgesetzt von den anderen Linien, sondern kreuzt mehrere der anderen Linien.

Der Grund für diesen Verlauf ist, dass den zugrundeliegenden Vorgaben (Mindestlänge von $0\,\mathrm{m}$, beliebig großer Kurvenradius) der komplette Streckenverlauf jedes beliebigen Datensatzes entspricht. Da im untersuchten Anwendungsfall nur Streckenabschnitte innerhalb von Straßenelementen (`<road>`) gefunden werden können (siehe Kapitel 3.1), resultieren diese Vorgaben in exakt einem identifizierten Streckenabschnitt je Straßenelement. Daher entspricht die gefundene Anzahl (siehe Tabelle 4.5) der Gesamtzahl an Straßenelementen im Datensatz (Tabelle 4.1): 248.

Abbildung 4.5 weist ebenfalls sechs Linien auf. Der Unterschied zu Abbildung 4.4

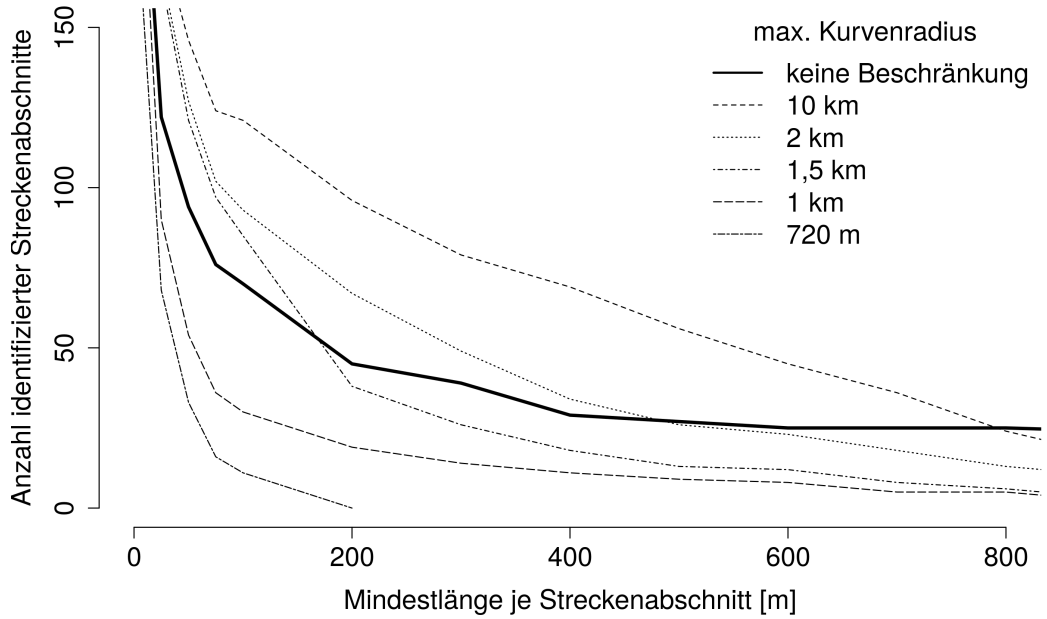


Abbildung 4.4: Anzahl identifizierter Streckenabschnitte nach Mindestlänge und maximalem Kurvenradius. Datengrundlage: Tabelle 4.5.

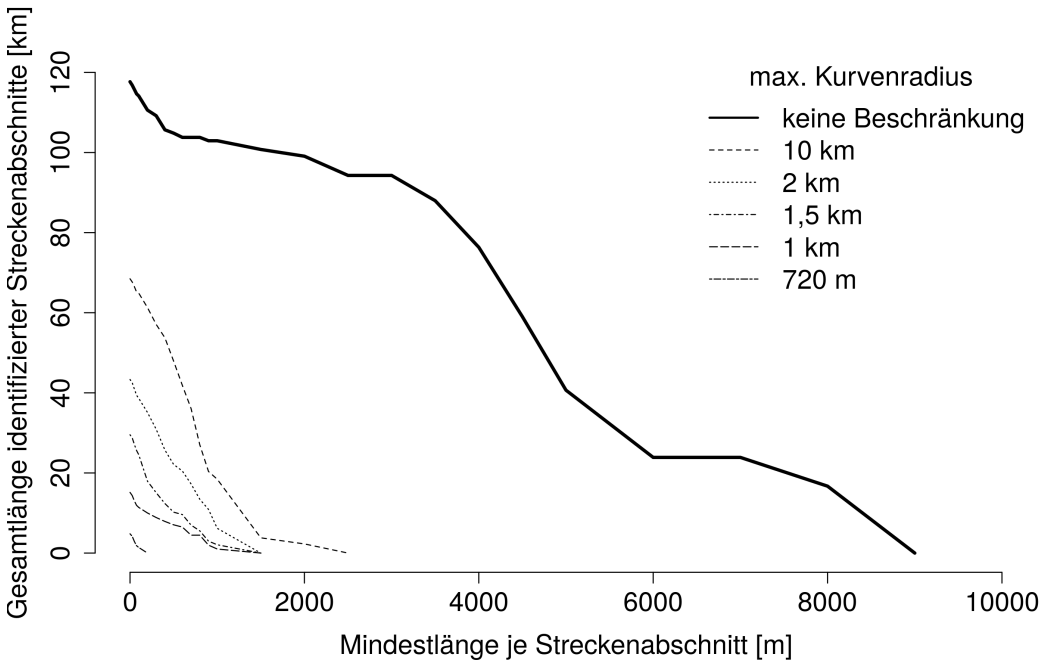
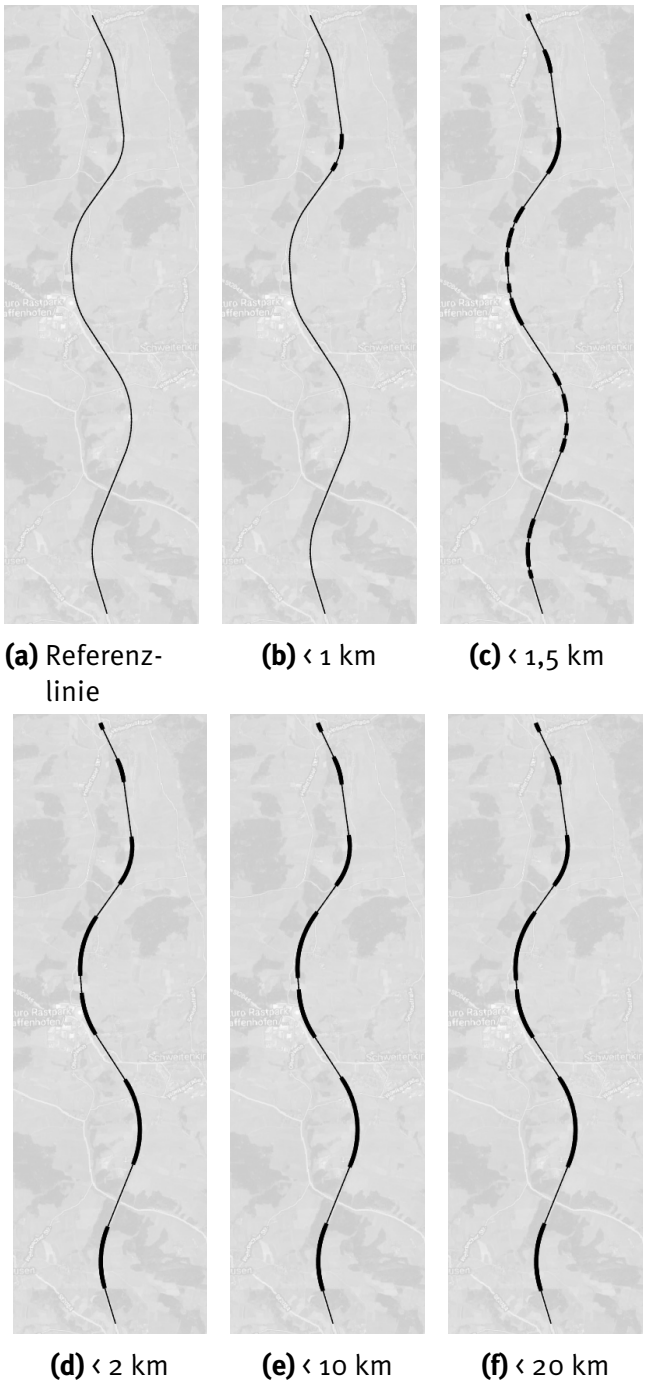


Abbildung 4.5: Gesamtlänge identifizierter Streckenabschnitte nach Mindestlänge und maximalem Kurvenradius. Datengrundlage: Tabelle 4.6.

besteht darin, dass nicht die Anzahl der gefundenen Streckenabschnitte, sondern deren Gesamtlänge gegen die Mindestlänge je Streckenabschnitt aufgetragen wird. Der Verlauf aller Linien entspricht den Erwartungen: monoton fallend und sich nicht kreuzend. Die Summe der Gesamtstrecken entspricht, genau wie bei der Orientierung, der Gesamtstrecke des Datensatzes bis auf $0{,}002\,\mathrm{km}$.

Der Satz vergleichender Karten in Abbildung 4.6 zeigt, welchen Einfluss die Änderung des Kurvenradiusmaximums darauf hat, welche Streckenabschnitte identifiziert werden. In sechs Karten des gleichen Ausschnitts wird vor dem Hintergrund der Referenzlinie eines etwa $10{,}5\,\mathrm{km}$ langen, aus drei Straßenelementen bestehenden, Ausschnitts aus dem OpenDRIVE-Datensatz gezeigt, welche Teile des Streckenverlaufs unterschiedlichen Ausprägungen des Parameters *maximaler Kurvenradius* entsprechen. Die hervorgerufenen Bereiche entsprechen den Ergebnissen, wenn keine Mindestlänge vor-

gegeben wird.



(a) Referenzlinie **(b)** ‹ 1 km **(c)** ‹ 1,5 km

(d) ‹ 2 km **(e)** ‹ 10 km **(f)** ‹ 20 km

Abbildung 4.6: Der etwa $10{,}5\,\mathrm{km}$ lange A-9-Abschnitt von Schernbuch (im Süden) nach Geisenhausen (im Norden). Zu sehen sind die OpenDRIVE-Referenzlinie (dünn gezeichnet) der Straßenelemente mit den `id`-Werten 2104000, 2105000 und 2106000. Hervorgehoben (dick gezeichnet) sind Streckenabschnitte mit der Teilgrafik entsprechendem Kurvenradius. Hintergrundkarte: Google Maps.

Tabelle 4.5: Anzahl identifizierter Streckenabschnitte nach Mindestlänge (d_{min}) und maximalem Kurvenradius.

d_{min} [m]	kein Max.	10 km	2 km	1.5 km	1 km	0.72 km
0	248	336	334	329	246	200
25	122	177	168	165	90	68
50	94	146	127	121	54	33
75	76	124	102	97	36	16
100	70	121	93	85	30	11
200	45	96	67	38	19	
300	39	79	49	26	14	
400	29	69	34	18	11	
500	27	56	26	13	9	
600	25	45	23	12	8	
700	25	36	18	8	5	
800	25	24	13	6	5	
900	24	16	10	3	2	
1000	24	14	5	2	1	
1500	22	2				
2000	21	1				
2500	19					
3000	19					
3500	17					
4000	14					
4500	10					
5000	6					
6000	3					
7000	3					
8000	2					
9000						

Tabelle 4.6: Gesamtstrecke identifizierter Streckenabschnitte nach Mindestlänge (d_{min}) und maximalem Kurvenradius.

d_{min} [m]	kein Max. [m]	10 km [m]	2 km [m]	1.5 km [m]	1 km [m]	0.72 km [m]
0	117669	68476	43329	29530	15144	4804
25	116815	67680	42538	28773	14367	4189
50	115742	66503	40940	27105	13001	2879
75	114651	65165	39393	25606	11909	1835
100	114121	64890	38639	24558	11396	1394
200	110588	61289	35108	17949	10008	
300	109185	57085	30914	15067	8872	
400	105670	53747	25716	12367	7897	
500	104835	47918	22129	10213	7055	
600	103780	41879	20490	9618	6493	
700	103780	36094	17257	6982	4465	
800	103780	27036	13431	5548	4465	
900	102940	20369	10906	2981	1978	
1000	102940	18435	6182	2074	1011	
1500	100762	3804				
2000	99081	2285				
2500	94282					
3000	94282					
3500	87966					
4000	76341					
4500	59011					
5000	40659					
6000	23882					
7000	23882					
8000	16699					
9000						

5 Diskussion und Ausblick

Durch die in Kapitel 4 vorgestellten Ergebnisse kann gezeigt werden, dass es im Rahmen des betrachteten Anwendungsfalls – logische PEGASUS-Szenarien und Streckendaten im OpenDRIVE-Format – möglich ist, Realdaten-Streckenabschnitte automatisiert und parameterbasiert zu identifizieren.

Es stellt sich jedoch heraus, dass es im Anwendungsfall von Parameter zu Parameter unterschiedlich gut umzusetzen ist. Die Nutzung nicht standardisierter `<userData>`-Elemente zur Spezifikation von Fahrbahnmarkierungen erschwert beispielsweise eine allgemeingültige Implementation. Auch finden sich in den Realdaten keine perfekt geraden Abschnitte, wie von PEGASUS-Szenarien gefordert werden könnte.

Im Folgenden werden die mit dem Anwendungsfall einhergehenden Annahmen und Einschränkungen diskutiert sowie zahlreiche Anknüpfungspunkte, Verfeinerungs- und Weiterentwicklungsmöglichkeiten genannt.

5.1 Grenzen und Einschränkungen

Als erste Verknüpfung von logischen PEGASUS-Szenarien und Streckendaten im OpenDRIVE-Format wurde in dieser Arbeit ein Anwendungsfall untersucht, der einigen Einschränkungen unterliegt: einerseits vereinfachende Annahmen, um den Rahmen der Untersuchung nicht ausarten zu lassen, und andererseits die formatinhärent unterschiedlichen Detailgrade der Daten. Zuletzt mussten einige Grenzwerte festgelegt werden, deren Festlegung an anderer Stelle genauer untersucht werden sollte.

Vereinfachende Annahmen

Es wurden mehrere vereinfachende Annahmen getroffen. Darunter hervorstechend die, dass die Fahrstreifenzahl genügt, um festzustellen, ob ein Streckenabschnitt der Vorgabe eines Regelquerschnitts entspricht (Kapitel 3.2.1). Für eine höhere Genauifkeit müssten zusätzlich die Breite der Fahrstreifen sowie das Vorhandensein und die Breiten des Banketts sowie des Standstreifens berücksichtigt werden. Umfangreicher wäre die Erweiterung auf Spezial-Regelquerschnitte für Brücken und Tunnel sowie ältere Regelquerschnitte (Kapitel 2.1.1), welche in dieser Arbeit durch die Fokussierung auf PEGASUS ausgeklammert wurden (Kapitel 3.1).

Erwähnenswert ist außerdem die fehlende Berücksichtigung der Leitplankenposition relativ zur Fahrbahn (siehe Kapitel 3.2.1). Diese Information ist allerdings im OpenDRIVE-Format enthalten und ließe sich mit weiterem Aufwand extrahieren.

Die Ableitung eines als konstant angenommenen Kurvenradius je Geometrieelement ist vereinfachend und kann besser durchgeführt werden. Etwa feiner auflösend auf Basis einzelner Punkte, statt den kompletten Linienzug zu betrachten, oder zuzüglich einer Glättung, beispielsweise durch gleitenden Durchschnitten (*moving-average*-Verfahren).

Logische PEGASUS-Szenarien und hochgenaue OpenDRIVE-Realdaten

Die Fokussierung auf PEGASUS bringt die Einschränkung auf anschlussstellenfreie Autobahnabschnitte mit sich. Autobahnanschlussstellen, Landstraßen und städtische Straßennetzwerke sind damit ausgeklammert gewesen. Außerdem wurde sich die Voraussetzung der Anschlussstellenfreiheit der zu identifizierenden Streckenabschnitte und der günstigen Datenlage – lange `<road>`-Elemente zwischen Anschlussstellen – zunutze gemacht, um lediglich innerhalb dieser Straßenelemente zu suchen. Falls ein OpenDRIVE-Datensatz feingliedriger in Straßenelemente zerteilt ist oder Anschlussstellen nicht mehr ausgeschlossen sind, bedarf es einer Anpassung des Algorithmus. Diese Anpassung kann auf Grundlage der im OpenDRIVE-Format vorliegenden Vorgänger- und Nachfolger-Angaben (siehe Kapitel 2.3.1) funktionieren.

Aus der Betrachtung logischer Szenarien geht die Fokussierung auf Werte*bereiche* statt konkreter Parameterausprägungen einher. Die Suche nach konkreten Parameterausprägungen hat im Rahmen des untersuchten Anwendungsfalls für Parameterbereiche mit eng begrenztem, diskreten Wertebereich wie Straßenorientierung, Anzahl Fahrsteifen (als Regelquerschnitt-Surrogat) und dem Vorhandensein von Leitplanken funktioniert. Für andere Wertebereiche, wie dem des Kurvenradius, stellt die hochgenaue Natur von aus Sensordaten abgeleiteten OpenDRIVE-Daten eine Herausforderung dar, denn diese entsprechen nie exakt einer einfachen Parameterausprägung, sondern weisen stets minimale Schwankungen auf. Abhilfe könnten Toleranzgrenzen schaffen, in deren Rahmen Schwankungen ignoriert werden. Beispielsweise könnte mit einer Toleranzgrenze von nur wenigen Prozentpunkten eine Kurve mit einem zwischen $1,48$ und $1{,}52\,\mathrm{km}$ schwankenden Kurvenradius zu einer Kurve mit einem Kurvenradius von $1{,}5\,\mathrm{km}$ abstrahiert werden. Solche Toleranzgrenzen könnten für die Breitenberechnung von Fahrstreifen genutzt werden. Im Grunde entspricht die Verwendung allerdings nur der Implementierung zusätzlicher Parameter *minimaler Kurvenradius* oder *prozentuale Toleranz*.

Die Nutzung von `<userData>`-Elementen, um Bestandteile eines Straßenverlaufs zu spezifizieren, wie es im Anwendungsfall etwa für Fahrbahnmarkierungen der Fall war, wird vom entwickelten Algorithmus nicht unterstützt. Da die Struktur dieser Elemente zwar zwischen Auftraggeber (etwa dem DLR) und dem Vermessungsdienstleister festgeschrieben werden kann, aber dennoch keiner Standardisierung unterliegt, stellt die Verarbeitung derart kodierter Daten auch in Zukunft eine Herausforderung dar.

Die in dieser Arbeit untersuchten Parameter sind den ersten beiden Beschreibungsebenen von Szenarien (Straßenlevel, Leitinfrastruktur) zugeordnet. Innerhalb dieser Beschreibungsebenen gibt es weitere Parameter, die nicht untersucht wurden. Parameter der dritten Beschreibungsebene (vorübergehende Änderungen der ersten beiden Ebenen), die ebenfalls in OpenDRIVE abgebildet würden, wurde gar nicht untersucht.

Bestimmung von Grenzwerten

Bei der Umsetzung dieser Arbeit wurden mehrere Grenzwerte festgesetzt. So wird ein Streckenverlauf ab einem Kurvenradius von 10 km als Gerade interpretiert (Kapitel 3.2.1) und eine Lücke von bis zu einem 1 m zwischen zwei Leitplanken-Instanzen als nicht-existent betrachtet (Kapitel 3.2.1). Dabei wurden diese Werte zunächst als plausible Annahmen festgelegt. Eine genauere Festlegung geeigneter Parameter-Intervalle oder Grenzwerte steht aus.

5.2 Ausblick

Im Folgenden werden mögliche Anknüpfungspunkte für weitere Arbeiten gegeben. Einige beziehen sich auf PEGASUS beziehungsweise mögliche Nachfolgeprojekte. Andere sind unabhängiger und beschäftigen sich mit unabhängigen, allgemeineren Anwendungsfällen.

5.2.1 Einbindung in PEGASUS-Werkzeugkette

Für eine vollständige Einbindung in die PEGASUS-Toolchain zum Automatisierten Testen von Fahrfunktionen fehlen noch einige Puzzleteile. Insbesondere betrifft das die Schnittstellen zum vorherigen und zum nachfolgenden Arbeitsschritt der Werkzeugkette. Im Folgenden werden einige dieser noch fehlenden Schritte beleuchtet.

Da PEGASUS keine präzise Vorschrift für die Spezifikation logischer Szenarien liefert, wurden in dieser Arbeit einfache Konfigurationsdateien verwendet (siehe 3.2.3). Für eine Einbindung in die Werkzeugkette von PEGASUS bedarf es einer projektübergreifenden Festlegung dieser Schnittstelle. Diese könnte etwa auf einer exakten Verwendungsvorschrift für das ohnehin für die Festlegung der Szenarioparameter der vierten Beschreibungsebenene (Verkehrsdynamik) genutzte OpenSCENARIO-Format aufsetzen.

Damit identifizierte Realdatenabschnitte zusätzlich zu synthetischen Streckenabschnitten in der Simulationskette verwendet werden können, werden sie als eigenständige OpenDRIVE-Dateien benötigt. Daher braucht es einen Weg, um Abschnitte aus Open-

DRIVE-Quelldaten eines realen Straßennetzes anhand des Outputs der in dieser Arbeit entwickelten Software (siehe Kapitel 3.2.3) auszuschneiden.

Die Basis für diese Aufgabe könnte ein bereits bestehender projektinterner Webdienst des DLR sein. Dieser stellt den groben Streckenverlauf vorhandener OpenDRIVE-Daten als *Simple Features* über eine GeoServer[1]-Instanz zur Verfügung. Auf einer Weboberfläche kann mithilfe einer Bounding-Box ein Ausschnitt ausgewählt werden, für welchen dann die entsprechenden Straßenelemente (`<road>`) der zugrundeliegenden OpenDRIVE-Daten als neue, eigenständige OpenDRIVE-Datei zurückgegeben werden.

Straßenelemente in aus Vermessungen entstandenen OpenDRIVE-Daten verlaufen typischerweise von Anschlussstelle zu Anschlussstelle. Daher sind sie in der Regel mehrere Kilometer lang und enthalten mehrere Kurven, so dass sie größer/länger als die Abschnitte sind, die den logischen Szenarien entsprechen. Für eine Einbindung in die Werkzeugkette müssten `<road>`-Elemente daher automatisiert zerteilt und gegebenenfalls zusammengefügt werden. Das wird vom angesprochenen Webdienst allerdings nicht geleistet.

5.2.2 Zusätzliche Parameter und Verkehrsumgebungen

In dieser Arbeit wurde die Suche nur für ausgewählte Parameter von PEGASUS-Szenarien implementiert. Für eine Nutzung im hypothetischen Produktivbetrieb wäre eine Erweiterung um die restlichen Parameter der Szenario-Beschreibungsebenen 1 bis 3 notwendig. Insbesondere mit der dritten Beschreibungsebene beschäftigt sich Runde [29].

Ein weiterer Anknüpfungspunkt ist die Überwindung der Beschränkung auf anschlussstellenfreie Autobahnabschnitte. Als Ausgangspunkt für das Fernziel vom Identifizieren von städtischen Straßennetzten nach bestimmten Vorgangen könnte Sämanns Arbeit [28] zum Kartenabgleich (*map matching*) dienen.

[1] eine auf Geodaten spezialisierte Serversoftware: `http://geoserver.org`

5.2.3 Ähnlichkeit mit konkreten Szenarien

Die Vermessung realer Streckenabschnitte bietet immer nur eine Approximation an die Realität, so dass selbst in der Realität mathematisch perfekt verlaufende Streckenabschnitte nicht nach einer Vermessung in einer Ansammlung von Polynom-Funktionen statt in einem – beispielsweise – perfekten Kreisbogen resultieren (siehe Kapitel 2.3.3).

Da reale Straßen nie ideal-mathematisch perfekte Verläufe haben, lässt sich ausschließen, perfekt auf konkrete Szenarien passende Streckenverläufe im realen Straßennetz zu finden. Denkbar ist aber, Streckenverläufe nicht nur auf Kompatibilität mit logischen Szenarien zu prüfen, sondern auf *Ähnlichkeit* mit *konkreten* Szenarien. Hierfür bräuchte es ein quantitatives Ähnlichkeitsmaß, welches sowohl die Geometrie eines Streckenverlaufs als auch weitere Attribute wie das Vorhandensein von Leitplanken einschließt. Einige Ideen hierfür liefert ebenfalls Sämann [28].

5.2.4 Alternative Datenquellen

Die Nutzung anderer Datenquellen und -formate als OpenDRIVE ist gut denkbar. OpenStreetMap[2] (OSM) liefert eine umfangreiche Datenbasis nicht nur für Autobahnen.

Viele Eigenschaften von Streckenabschnitten werden jedoch nicht geometrisch, sondern attributbasiert abgespeichert. So wird die Fahrbahn einer Autobahn durch einen Linienzug abgebildet, der lediglich mit Attributen zu Leitplanken oder Fahrstreifenzahl angereichert werden kann. Solche Daten sind deutlich weniger präzise, aber einfacher zu durchsuchen.

Durch den Community-getriebenen Ansatz von OpenStreetMap schwankt die Aktualität und Qualität der Daten. Zumindest die Aktualität ist für die Simulation von Fahrfunktionen kaum relevant. Auch ein Streckenabschnitt aus dem letzten Jahr ist ein realistischer Streckenabschnitt und damit ein Mehrwert für die Simulation.

Eine weiteres denkbares Datenformat ist das NDS[3]-Format. Hierbei handelt es sich um ein standardisiertes Format, welches insbesondere für die Speicherung von Karten-

[2] https://www.openstreetmap.org/
[3] Navigation Data Standard

Tabelle 5.1: Vereinfachter Ausschnitt aus der Eventliste des Straßenelements mit dem `id`-Wert `2104000` aus dem untersuchten OpenDRIVE-Datensatz. Zu sehen sind einige den Parameter Straßenorientierung betreffende Events.

ROADID	S0	S1	LENGTH	ORIENTATION
2104000	0.0	52.6	52.6	0
2104000	52.6	130.3	77.6	0
2104000	130.3	226.6	96.3	0
2104000	226.6	300.6	74.0	0
2104000	300.6	376.6	75.9	0
2104000	376.6	449.3	72.6	0
2104000	449.3	501.6	52.3	0
2104000	501.6	540.9	39.3	0
2104000	540.9	587.3	46.3	0
2104000	587.3	633.3	46.0	-1
2104000	633.3	676.0	42.6	-1
2104000	676.0	708.3	32.3	-1

datenbanken für Navigationssysteme verwendet wird.

5.2.5 Parametrisierung von Realdaten

Wie bereits in Kapitel 3.2.1 angedeutet, bieten die als Zwischenprodukt des in dieser Arbeit entwickelten Identifikationsalgorithmus anfallenden Eventlisten einen guten Ausgangspunkt, um Realdaten zu parametrisieren.

Tabelle 5.1 zeigt einen Ausschnitt aus der Eventliste eines Straßenelements des untersuchten Datensatzes. Es ist ersichtlich, dass das Straßenelement mit einem geraden Abschnitt (Orientierung: 0) beginnt und an der Stelle $s = 587,3$ in eine Rechtskurve (Orientierung: -1) übergeht.

Mit derart aufbereiteten Informationen ließen sich, in Kombination mit der Arbeit von Scholz und Idrobo [19] und einem Softwarewerkzeug zum Zerschneiden von Straßenelementen (siehe Kapitel 5.2.1) mit hochwertigen Informationen angereicherte,

OGC-konforme Geodaten erzeugen. Auf dieser Grundlage wird eine Zusammenführung mit Daten aus Quellen deutlich einfacher. Auch könnten sie als Spende an das OpenStreetMap-Projekt gehen, um einer größeren Öffentlichkeit zugänglich gemacht zu werden.

6 Fazit

Diese Arbeit hat anhand eines Beispiels gezeigt, dass die automatisierte und parameterbasierte Identifikation von Realdaten-Streckenabschnitten möglich ist. Es wurden realdatenformat- und parameterabhängige Schwierigkeiten in der Durchführbarkeit festgestellt, von denen für die meisten Lösungsvorschläge präsentiert oder vorgeschlagen wurden. Unter dem Stichwort *Parametrisierung von Realdaten* wurde in Ausblick gestellt, Realdaten nicht nur parameterbasiert durchsuchen zu können, sondern in eine parameterbasierte Repräsentation umzuwandeln.

Literaturverzeichnis

[1] Bundesministerium für Verkehr und digitale Infrastruktur. *Bericht der Ethik-Kommission Automatisiertes und Vernetztes Fahren*. 2017. URL: https://www.bmvi.de/SharedDocs/DE/Publikationen/DG/bericht-der-ethik-kommission.pdf (besucht am 12. 07. 2019).

[2] Keen Security Lab. *Experimental Security Assessment of BMW Cars: A Summary Report*. 2018. URL: https://keenlab.tencent.com/en/whitepapers/Experimental_Security_Assessment_of_BMW_Cars_by_KeenLab.pdf (besucht am 12. 07. 2019).

[3] Daniel Lipinski. „How good is good enough“. PEGASUS-Abschlussveranstaltung, Ehra-Lessien, Deutschland. 2019. URL: https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/PEGASUS_Abschlussveranstaltung_Wie_gut_ist_gut_genug.pdf (besucht am 12. 07. 2019).

[4] Walther Wachenfeld und Hermann Winner. „Die Freigabe des autonomen Fahrens“. In: *Autonomes Fahren*. 2015. URL: https://www.researchgate.net/publication/283326665 (besucht am 12. 07. 2019).

[5] Wolfgang Konen. *Spiele, Simulation und dynamische Systeme*. URL: http://www.gm.fh-koeln.de/~konen/WPF-Spiele/materialien/WPF-Teil1/OekoSim0/OeSi1.htm (besucht am 12. 07. 2019).

[6] Fabian Schuldt. „Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen“. Diss. Apr. 2017. URL: http://uri.gbv.de/document/gvk:ppn:884860981 (besucht am 12. 07. 2019).

[7] Pascal Marcel Minnerup. „An Efficient Method for Testing Autonomous Driving Software against Nondeterministic Influences“. Diss. Technische Universität München, 2017. URL: https://nbn-resolving.org/urn/resolver.pl?urn:nbn:de:bvb:91-diss-20171220-1353122-1-5 (besucht am 12. 07. 2019).

[8] *Forschungsprojekt Pegasus*. Bundesministerium für Wirtschaft und Energie. URL: https://www.pegasusprojekt.de (besucht am 12. 07. 2019).

[9] Forschungsgesellschaft für Straßen- und Verkehrswesen. Arbeitsgruppe. *Richtlinien für die Anlage von Autobahnen*. FGSV-Verlag, 2008. URL: https://books.google.de/books?id=EG8fYAAACAAJ (besucht am 12. 07. 2019).

[10] Gerrit Bagschik, Till Menzel, Andreas Reschka und Markus Maurer. „Szenarien für Entwicklung, Absicherung und Test von automatisierten Fahrzeugen“. In: *11. Workshop Fahrerassistenzsysteme und automatisiertes Fahren* (2017). URL: https://www.researchgate.net/publication/316452171 (besucht am 12. 07. 2019).

[11] Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt und Markus Maurer. „Definition der Begriffe Szene, Situation und Szenario für das automatisierte Fahren“. In: Sep. 2015. URL: https://www.researchgate.net/publication/283726191 (besucht am 12. 07. 2019).

[12] „Szenarienbeschreibung“. 2017. URL: https://www.pegasusprojekt.de/files/tmpl/PDF-HZE/04_Szenarienbeschreibung.pdf (besucht am 12. 07. 2019). Poster, ausgestellt auf der PEGASUS-Halbzeitveranstaltung, Aachen, Deutschland.

[13] „Scenario Formats“. 2019. URL: https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/14_Scenario-Formats.pdf (besucht am 12. 07. 2019). Poster, ausgestellt auf der PEGASUS-Abschlussveranstaltung, Ehra-Lessien, Deutschland.

[14] *OpenDRIVE®. Format Specification, Rev. 1.4*. 4. Nov. 2015. URL: http://www.opendrive.org/docs/OpenDRIVEFormatSpecRev1.4H.pdf (besucht am 12. 07. 2019).

[15] Andreas Richter und Michael Scholz. „The Surveyor's Guide to the Driving Simulation: Developing and Evaluating of Guidelines for Straightforward Road Surveying for Driving Simulator Databases for Development and Test of Driver Assistance and Automation Systems". In: *Driving Simulation Conference 2016 Europe VR*. Hrsg. von Andras Kemeny, Frederic Merienne, Florent Colombet und Stephane Espie. Driving Simulator Association, Sep. 2016, S. 119–122. URL: https://elib.dlr.de/102470/ (besucht am 12. 07. 2019).

[16] Andreas Richter und Michael Scholz. „Deploying Guidelines and a Simplified Data Model to Provide Real World Geodata in Driving Simulators and Driving Automation". In: *Transportation Research Part F: Traffic Psychology and Behaviour*. Transportation Research Part F: Traffic Psychology and Behaviour Editorial Board (Mai 2017). Hrsg. von T. Lajunen, T. Özkan und B. E. Porter. URL: https://elib.dlr.de/110094/ (besucht am 12. 07. 2019).

[17] Andreas Richter, Michael Scholz, Hartmut Friedl, Thomas Ruppert und Frank Köster. „Challenges and experiences in using heterogeneous, geo-referenced data for automatic creation of driving simulator environments". In: *Simulation-Transactions of the Society for Modeling and Simulation International*. SIMULATION 92.5 (Apr. 2016). Hrsg. von Mikel D. Petty und Gabriel Wainer, S. 437–446. URL: https://elib.dlr.de/109231/ (besucht am 12. 07. 2019).

[18] Ulf Noyer, Andreas Richter und Michael Scholz. „Generation of Highway Sections for Automated Test Case Creation". In: *Driving Simulation Conference Europe 2018 VR*. 2018, S. 171–174. URL: https://elib.dlr.de/117113/ (besucht am 12. 07. 2019).

[19] Michael Scholz und Ana Maria Orozco Idrobo. „Supporting the Implementation of Driving Simulator Environments Through Established GIS Approaches by Extending the Geospatial Data Abstraction Library (GDAL) with OpenDRIVE".

In: *Proceedings of the Driving Simulator Conference 2017 Europe VR*. Sep. 2017, S. 51–54. URL: https://elib.dlr.de/110123/ (besucht am 12. 07. 2019).

[20] Michael Scholz und Ana Maria Orozco Idrobo. *OpenDRIVE Driver for OGR*. https://github.com/DLR-TS/gdal. 2017. (Besucht am 12. 07. 2019).

[21] Prof. Dr. Thomas Form. „Ziele und Arbeitsinhalte“. 2017. Vortrag, gehalten auf der PEGASUS-Halbzeitveranstaltung, Aachen, Deutschland.

[22] „J3016. Taxonomy and Definitions for Terms Related to Driving Automation systems for On-Road Motor Vehicles“. 2016.

[23] „Der Autobahn-Chauffeur“. 2017. URL: https://www.pegasusprojekt.de/files/tmpl/PDF-HZE/03_Der-Autobahn-Chauffeur.pdf (besucht am 12. 07. 2019). Poster, ausgestellt auf der PEGASUS-Halbzeitveranstaltung, Aachen, Deutschland.

[24] Dr. Helmut Schittenhelm. „Testen von L3-Systemen – ein Überblick über die aktuelle PEGASUS-Vorgehensweise“. 2017. URL: https://www.pegasusprojekt.de/files/tmpl/HZE-Vortraege/04_PEGASUS_HZE_Testen%20von%20L3%20Systemen.pdf (besucht am 12. 07. 2019). Vortrag, gehalten auf der PEGASUS-Halbzeitveranstaltung, Aachen, Deutschland.

[25] „Testkonzept und Testfallzuordnung“. 2017. URL: https://www.pegasusprojekt.de/files/tmpl/PDF-HZE/08_Herausforderungen-eines-szenario-basierten-Ansatzes.pdf (besucht am 12. 07. 2019). Poster, ausgestellt auf der PEGASUS-Halbzeitveranstaltung, Aachen, Deutschland.

[26] „Übergreifende Testinfrastrukturen / Karten“. 2017. URL: https://www.pegasusprojekt.de/files/tmpl/PDF-HZE/08_Herausforderungen-eines-szenario-basierten-Ansatzes.pdf (besucht am 12. 07. 2019). Poster, ausgestellt auf der PEGASUS-Halbzeitveranstaltung, Aachen, Deutschland.

[27] Haozhou Wang, Han Su, Kai Zheng, Shazia Sadiq und Xiaofang Zhou. „An Effectiveness Study on Trajectory Similarity Measures“. In: *Proceedings of the Twenty-Fourth Australasian Database Conference - Volume 137*. ADC '13. Adelaide, Australia: Australian Computer Society, Inc., 2013, S. 13–22. URL: http://dl.acm.org/citation.cfm?id=2525416.2525418 (besucht am 12. 07. 2019).

[28] Robert Sämann. „Bestimmung einer Bewertungsmetrik zum Vergleich digitaler Straßennetze für Verkehrsflusssimulation und Routing von Einsatzkräften“. Masterarbeit. Leibniz Universität Hannover, Okt. 2014. URL: https://elib.dlr.de/91532/ (besucht am 12. 07. 2019).

[29] Marius Runde. „Strategies for Automated Updating of Road Networks for Driving Simulations“. Magisterarb. Westfälische Wilhelms-Universität Münster, Aug. 2017. URL: https://elib.dlr.de/114809/ (besucht am 12. 07. 2019).

Erklärung

Hiermit versichere ich, die vorliegende Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt, sowie die Zitate deutlich kenntlich gemacht zu haben.

Münster, 12. Juli 2019

(Peter Konopatzky)