Entwicklung einer Funktion zur spurgenauen Lokalisierung basierend auf visuellen Informationen

Taihao Li[1*], Xinhai Xu[1], Marian Göllner[1], Sven Jacobitz[1],Xiaobo Liu-Henke[1]

[1]Fachgruppe Regelungstechnik und Fahrzeugmechatronik, Hochscule Ostfalia, Salzdalumer 46/48, 38302 Wolfenbüttel, Deutschland; *ta.li@ostfalia.de*

Kurzfassung. Die Einführung neuer digitaler Technologien treibt den gesellschaftlichen und wirtschaftlichen Wandel voran, insbesondere im Bereich der Mobilität, die durch autonomes Fahren in cyber-physischen Systemen (CPS) ermöglicht wird. Die Realisierung autonomer Fahrsysteme erfordert Fahrzeuge in der Lage, in komplexen Straßenumgebungen eine spurgenaue Lokalisierung und eine spurgenaue Lokalisierung anderer Verkehrsteilnehmerdurchzuführen. In dieser Arbeit wird eine Funktionalität zur spurgenaueren Selbstlokalisierung und zur genaueren Lokalisierung anderer Verkehrsteilnehmer entwickelt, die auf visuellen Informationen basiert. Diese ermöglicht es dem Fahrzeug, Straßeninformationen und andere Verkehrsteilnehmer zu identifizieren. Die Funktionen werden in einem virtuellen Testszenario validiert.

Einleitung

Derzeit befinden sich Gesellschaft und Wirtschaft in einem Transformationsprozess aufgrund des Einsatzes neuer digitaler Technologien. Eine bedeutende Technologie für zukünftige Mobilität ist das autonome Fahren in der vernetzten CPS-Umgebung [1]. Die Society of Autonomous Engineers (SAE) hat sechs verschiedene Stufen des autonomen Fahrens (L0-L5) spezifiziert [2]. Das Ziel des autonomen Fahrens (L5) besteht darin, dass Fahrzeuge ohne Eingriff des Fahrers vollständig autonom die Wahrnehmung, Entscheidungsfindung und Ausführung übernehmen, die ursprünglich dem Fahrer oblag. Aus [3] ist bewusst, dass Hochpräzise Navigations und Lokalisierung eine der Schlüsseltechnologien für autonome Fahrzeuge sind, wobei die spurgenaue Lokalisierung eine Grundvoraussetzung für diese Lokalisierung darstellt. Eine weit verbreitete Methode zur spurgenauen Positionierung ist die Verwendung von GNSS Echtzeit-Kinematik (RTK), die jedoch häufig aufgrund schlechter Satellitengeometrie ausfällt und zu erheblichen Fehlern führt. Darüber hinaus führen Unterbrechungen des Funksignals durch Wolkenkratzer in städtischen Gebieten zu Problemen [4]. Eine Alternative für die genaue Lokalisierung ist das Matching mit einer Karte, der auf Sensordaten und früheren Karten basiert. Durch das Abgleichen von Lidar-Daten mit Punktwolkenkarten kann eine hochpräzise Ortung erreicht werden [5]. Aber mit dieser Methode nimmt oft viel Speicherplatz in Anspruch und erfordert häufige Aktualisierungen der Karte.

Ein Ansatz ist die Bestimmung der Fahrzeugposition durch visuelle Erkennung der Fahrspur und der benachbarten Fahrspuren. Darüber hinaus ist es für eine effiziente Routenplanung auf Fahrstreifenebene erforderlich, dass die Fahrzeuge Hindernisse und Verkehrsteilnehmer auf anderen Fahrstreifen wahrnehmen. Die Objektlokalisierung sollte spurgenau erfolgen, um die Routeplanung zu optimieren. Daher wird in dieser Arbeit die Bilder aus der Kamera verwendet, um Fahrspur zu erkennen. Dadurch wird eine Spurgenaue Lokalisierung des Ego-Fahrzeugs sowie eine Spurgenaue Lokalisierung anderer Verkehrsteilnehmer ermöglicht.

Der weitere Beitrag ist wie folgt gegliedert: zunächst wird in Abschnitt eins der strukturierte, modellbasierte und verifikationsorientierte Rapid Control Prototyping (RCP)-Entwicklungsprozess ausführlich erläutert. Anschließend werden in Abschnitt zwei die aktuellen Fortschritte und Herausforderungen bei der erforderliche Technologie dargestellt. Der dritte Abschnitt umfasst die Konzeption für die Lokalisierungsfunktion. In Abschnitt vier wird schließlich den Entwurf der Lokalisierung vorgestellt. Die Ergebnisse der spurgenauen Lokalisierung werden in Abschnitt fünf dargestellt. Abschließend erfolgt eine Zusammenfassung der Arbeit und ein Ausblick auf zukünftige For-

schungsarbeiten.

1 Methodik

Der durchgängig modellbasierte und verifikationsorientierte Funktionsentwurf und die Absicherung vernetzter mechatronischer Systeme nach [6] hat sich in zahlreichen Anwendungen in Forschung und Industrie als zeit- und kosteneffizient erwiesen. Abbildung 1 zeigt den modellbasierten mechatronischen Entwicklungskreislauf, der zur Absicherung Model-in-the-Loop (MiL)-, Software-in-the-Loop (SiL)- und Hardware-in-the-Loop (HiL)-Simulationen sowie die Echtzeitrealisierung durch Prototypen umfasst.

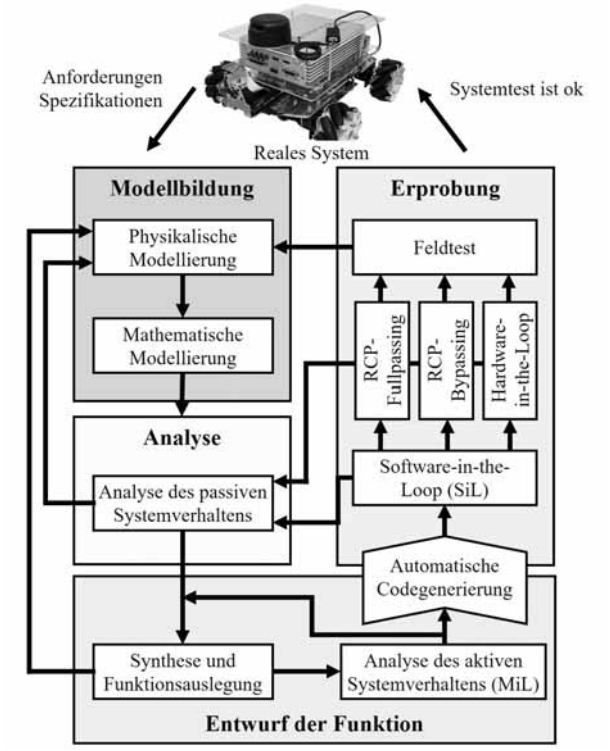

Abbildung 1: Mechatronischer Entwicklungskreislauf.

Der Entwurfsprozess beginnt mit der Modellbildung basierend auf dem realen System, welches gemäß der Anforderungen reduziert bzw. vereinfacht wird, sodass sich zunächst ein physikalisches Modell ergibt. Dieses wird mithilfe physikalischer Gesetzmäßigkeiten in ein mathematisches Modell überführt, welches wiederum bspw. in Form von Signalflussplänen im Rechner abgebildet und mithilfe von CAE-Werkzeugen und entsprechender Numerik simuliert werden kann. Der Modellbildungsprozess umfasst Messungen am realen System. Dabei werden die Parameter des mathematischen Modells identifiziert und die Simulation validiert. Eine anschließende Analyse der Simulationsergebnisse lässt Schlussfolgerungen über die grundlegenden statischen und dynamischen Eigenschaften des realen Systems zu, auf dessen Grundlage die Konzeption der Funktionen erfolgt. Es werden sowohl die Funktionsarchitektur als auch Ansätze zu dessen Auslegung und Optimierung festgelegt. Komplexe Funktionen werden zudem nach dem verallgemeinerten Kaskadenprinzip in hierarchische Teilfunktionen zerlegt um die Funktionskomplexität zu reduzieren und so den Auslegeprozess handhabbar zu machen. Dazu werden bereits in einem frühen Entwicklungsstadium Hard- und Softwareanforderungen berücksichtigt und Schnittstellen für die funktionsübergreifende Kommunikation definiert. Der Erprobungsprozess wird parallel zur Entwicklung durchgeführt. Wenn eine Teilfunktion entwickelt worden ist, wird diese getestet und analysiert.

In dieser Arbeit wird mit Hilfe dieser Methode zunächst die Anforderungen werden bestimmt. Durch die Analyse des physikalischen Modells, sowie der Mathematik, wird die Funktion modelliert. Jeder Teil der Funktion wird entsprechend getestet.

2 Stand des Wissens

Mit der Entwicklung des Deep Learning wurden verschiedene auf Deep Learning basierende Algorithmen für die Aufgabe der Fahrspurerkennung vorgeschlagen [7], von denen die meisten gefaltete Neuronales Netze verwenden [8]. In [9] wird eine Technik vorgeschlagen, die neuronale Netze mit dem langes Kurzzeitgedächtnis (convLSTM) kombiniert, um die zukünftige Trajektorie eines Fahrzeugs vorherzusagen. In [10]wird ein auf semantischer Segmentierung basierendes Kodierungs-/Dekodierungsnetzwerk vorgeschlagen. Nach dem Downsampling des Bildes wird ResNet-50 verwendet, um die gewünschten Merkmale zu extrahieren, die dann mit dem Unet-Modell decodiert und upgesamplet werden. In [11]wird ein Netzwerk mit semantischer Segmentierung und optischer Flussschätzung kombiniert. Die semantische Segmentierung wird für Schlüsselbilder verwendet. Unkritische Frames werden mit Hilfe eines Netzwerks zur Schätzung des optischen Flusses verfolgt und schließlich werden die Fahrspuren mit Hilfe von Clustering (Density-Based Spatial Clustering of Applications with

Noise) identifiziert. Das Modell LanetNet [8] zeichnet sich durch seine hohe Leistungsfähigkeit aus. Es ist ein End-to-End-Deep-Learning-Modell, das die gleichzeitige Erkennung und Segmentierung von Fahrspurlinien, ein Merkmalsextraktionsnetzwerk und ein integriertes Clusternetzwerk umfasst. Das Merkmalsextraktionsnetz verwendet eingefaltete Neuronales Netz, um Merkmale aus dem Bild zu extrahieren. Das Clusternetzwerk hingegen gruppiert die extrahierten Merkmale, um die verschiedenen Fahrspurlinien zu segmentieren. Zu den Vorteilen gehören eine höhere Echtzeitleistung und Robustheit. Im LaneNet-Modell werden häufig vortrainierte Modelle als Kodierer verwendet. Der Vorteil von Pre-Trainingsmodellen besteht darin, dass sie die Merkmalsextraktion verbessern und den Trainingsprozess bis zu einem gewissen Grad beschleunigen können. VGG16 [12], ResNet [13] und BiSeNet V2 [14] sind häufig verwendete Pre-Trainingsmode.

Die Technik der Objekterkennung löst das Problem der Klassifizierung und Lokalisierung von Objekten in einem Bild [15]. Die Anwendung dieser Technik legt den Grundstein für die automatische Analyse und das Verständnis komplexer Szenario. Ihre Effektivität wirkt sich direkt auf die Leistung verschiedener KI-Anwendungen aus, z. B. auf die Objektverfolgung [16, 17].

Die klassische Objektierkennungsalgorithmus waren hauptsächlich vor dem Jahr 2008 verbreitet. Sie basierten auf manuell extrahierten Merkmalen und folgten in etwa folgendem Ablauf: Markierung von Kandidatenbereichen, Merkmalsextraktion, Erkennung und Klassifizierung. Repräsentative Detektoren sind der VJ (Viola-Jones) Detektor [18] und der DPM (Deformable Parts Model) Detektor [19]. Zu den Nachteilen traditioneller Objekterkennungsalgorithmen zählen geringe Genauigkeit, schwache Robustheit und langsame Verarbeitungsgeschwindigkeit. Mit der Entwicklung der künstlichen Intelligenz sind allmählich Objekterkennungsalgorithmen auf Basis von Deep Learning entstanden

Die auf Deep Learning basierenden Objekterkennungsalgorithmen lassen sich hauptsächlich in zwei Arten unterteilen: One-Stage- und Two-Stage-Verfahren. Das Two-Stage-Verfahren R-CNN [20], Fast R-CNN [21] und Cascade R-CNN [22] lösen dieses Problem, in dem sie die Bildregionen in kleine Regionen aufteilen und separat klassifizieren. Zuerst werden den Erkennungsbereich abgegrenzt, dann erfolgt die Merkmalsextraktion und Klassifizierung. R-CNN zeichnet sich durch die Verwendung von Faltungsnetzen zur Merkmalsextraktion aus, wodurch die Schwächen herkömmlicher Erkennungsmethoden ausgeglichen werden, und verwendet eine selektive Suche zur Abgrenzung vom Bereich von Interesse (ROI), wodurch der Rechenaufwand reduziert wird. Fast R-CNN baut auf R-CNN auf und verbessert die Erkennungsgeschwindigkeit durch die Einführung einer neuen Verlustfunktion und erreicht eine erste Implementierung der End-to-End-Erkennung [23]. Two-Stage-Verfahren zeichnen sich durch eine hohe Erkennungsgenauigkeit aus, sind jedoch langsamer und für Echtzeitanwendungen weniger geeignet. One Stage Verfahren, vertreten durch die YOLO-Serie [24], zeichnen sich dadurch aus, dass sie keine Begrenzung des Erkennungsbereichs benötigen und direkt die Wahrscheinlichkeit der Objektkategorie und die Positionsdaten liefern, was die Erkennungsgeschwindigkeit deutlich erhöht.

Für Algorithmen zur Zielverfolgung lassen sich die derzeit verwendeten Verfolgungsschritte in zwei Hauptteile unterteilen: 1. Bewegungsmodell und Zustandsschätzung, die hauptsächlich dazu dienen, die Position von Objekten in nachfolgenden Frames vorherzusagen; 2. die Assoziation neuer Frame-Erkennungsergebnisse mit der aktuellen Trajektorie. Die derzeit führenden Tracking-Algorithmen sind DeepSORT [25] und Bytetrack [26]. Ersterer ist eine Weiterentwicklung des SORT-Algorithmus und verwendet die Mahalanobis-Distanz und Erscheinungsinformationen, um die Tracking-Frames des aktuellen Frames mit den Erkennungs-Frames abzugleichen. Allerdings sind der Rechenaufwand und die Verzögerung relativ hoch, so dass es für Anwendungen, die eine sofortige Reaktion erfordern, ungeeignet sein kann. Bytetrack verfügt über eine starke Assoziationsstrategie, die durch ein Sekundärer-Matching von Trajektorien mit hohen und niedrigen Vertrauensrahmen sowohl die Genauigkeit beibehält als auch die Geschwindigkeit erhöht und eine gute Robustheit bietet.

3 Konzeption

Im vorliegenden Abschnitt wird eine Konzeption für die spurgane Lokalisierung dargestellt. Abbildung 2 zeigt die Konzeption des der Spurgenaue Lokalisierungsfunktion. Um eine spurgenaue Lokalisierung zu ermöglichen, muss die Funktion folgende Anforderungen erfüllen: Das Fahrzeug kann Fahrspuren erkennen und die Anzahl der Fahrspuren bestimmen, auf denen es

sich befindet. Das Fahrzeug kann andere Verkehrsteilnehmer erkennen und die Fahrspuren der anderen Verkehrsteilnehmer bestimmen.

In dieser Arbeit kann ein Video V als eine chronologisch geordnete Sammlung von Einzelbildern betrachtet werden, wobei jedes Einzelbild als Matrix dargestellt und mit $\underline{\underline{B}}(t)$ bezeichnet wird. Außerdem wird die Bildrate des Videos als r bezeichnet wird.

- Erkennung der Fahrspur: Der Hauptzweck dieses Teils besteht darin, die Fahrspur-Informationen in den Daten zu extrahieren. Die jede Spurtrennlinie $l_i(x)$ können durch ein Polynom 1 dargestellt. i bedeutet die Nummer der Spurtrennlinie und x bedeutet den und der Koordinatenwert der Fahrspur in der Fahrtrichtung

$$l_i(x) = a_n x^n + \ldots + a_1 x + a_0 \tag{1}$$

  Der Gesamtrahmen dieses verwendeten Modells ist als Kodierungs-/Dekodierungsstruktur konzipiert, die für die binäre Segmentierung und die Instanzsegmentierung zuständig ist. Der Kodierer ist für die Merkmalsextraktion und der Dekodierer für die Verlustberechnung bei der binären Segmentierung und der Instanzsegmentierung zuständig. Der binäre Segmentierungsbild $\underline{\underline{B}}_b(t)$ und der Instanzsegmentierungsbild $\underline{\underline{B}}_i(t)$ wird durch das Modell erstellt. Bei der binären Segmentierung handelt es sich um eine semantische Segmentierung in zwei Kategorien (Fahrspur/Hintergrund); bei der Instanzsegmentierung wird die Cluster-Konzept verwendet, um jede Fahrspurlinie zu klassifizieren. Um die Genauigkeit der Anpassung der Fahrspurlinien zu sichern, muss das Rauschen im Binärbild verarbeitet werden. Das verarbeitete binäre Segmentierungsbild und das Instanzsegmentierungsbild werden als Eingaben verwendet, um ihre Fahrspurlinienmerkmale zu extrahieren. Die erhaltene Ausgabe sind die Koordinaten des Fahrspurlinienpunktsatzes $C_i(t)$. Schließlich werden die Fahrspurpolynome durch Kombination der Clustering-Ergebnisse angepasst, um ein Polynom für jede Fahrspurlinie zu erhalten.
- Lokalisierung: Das Hauptziel dieses Teils besteht darin, Verkehrsinformationen aus den Daten zu extrahieren, die zur Identifizierung von Verkehrsteilnehmerkategorien wie Fahrzeugen, Fußgängern, Fahrrädern usw. erforderlich sind. Sobald die Kategorie erkannt ist, muss das erkannte Objekt verfolgt werden. Es ist auch erforderlich, für jedes erkannte Fahrzeug eine eindeutige Identifikationsnummer id festzulegen und Informationen über die Position $\underline{P}_{id}(t)$ des Fahrzeugs sowie Informationen über die Fahrspur $l_{id}(t)$, auf der das Fahrzeug fährt, aufzuzeichnen.

4 Entwurf

In diesem Kapitel wird der Entwurf der Lokalisierung vorgestellt. In dieser Arbeit wird das Design der Lokalisierung beschrieben. In diesem Kapitel wird ein Video als Eingabe verwendet. Das Video zeigt die Verkehrsinformationen der Fahrspuren aus der Ego-Perspektive. Wie bereits erwähnt, umfasst die Informationsauswertung sowohl die Identifizierung von Fahrspurlinien als auch die Lokalisierung.

4.1 Erkennung der Fahrspur

In diesem Modell wird das Kodiermodell BiSeNetV2 verwendet. Die Aufgabe dieses Teils ist die Merkmalsextraktion, die aufgrund der parallelen Verzweigungsstruktur von BiSeNetV2 rechnerisch effizient ist. Nach der Dekodierung werden die binäre Abbildung und die Instanzabbildung erhalten. Dann wird die morphologische Schließungsoperation an der binären Abbildung durchgeführt, um die kleinen Hohlräume im Bild zu beseitigen und die Bildgrenzen zu glätten, um das Ergebnis der binären Segmentierung zu optimieren und zu bereinigen und ein klareres und vollständigeres Ergebnis der Fahrspursegmentierung zu erhalten. Danach wird mit der Verarbeitung der Segmentierungskarte begonnen: Zunächst werden die verarbeitete binäre Karte und die Instanzabbildung als Eingaben verwendet, um ihre Fahrspurmerkmale zu extrahieren, und die erhaltene Ausgabe sind die Koordinaten der Fahrspurpunktsätze.Das DBSCAN-Clustering wird für die Koordinaten der Fahrspurlinienpunkte durchgeführt, um die Anzahl der Cluster, die Clusterbeschriftungen und die Clusterzentren zu ermitteln. Um die Genauigkeit der Fahrspurberechnung zu gewährleisten, müssen Fahrspurpolynome durch Kombination der Clustering-Ergebnisse und Fahrspurkoordinaten angepasst werden, um die spezifischen Punktkoordinaten jeder Fahrspur zu erhalten. Das Verfahren wird durch Prozedur 1 dargestellt.

4.2 Lokalisierung

Die Lokalisierung erfordert zunächst eine genaue Erkennung der Fahrzeuge im Video. Da diese Aufgabe ei-

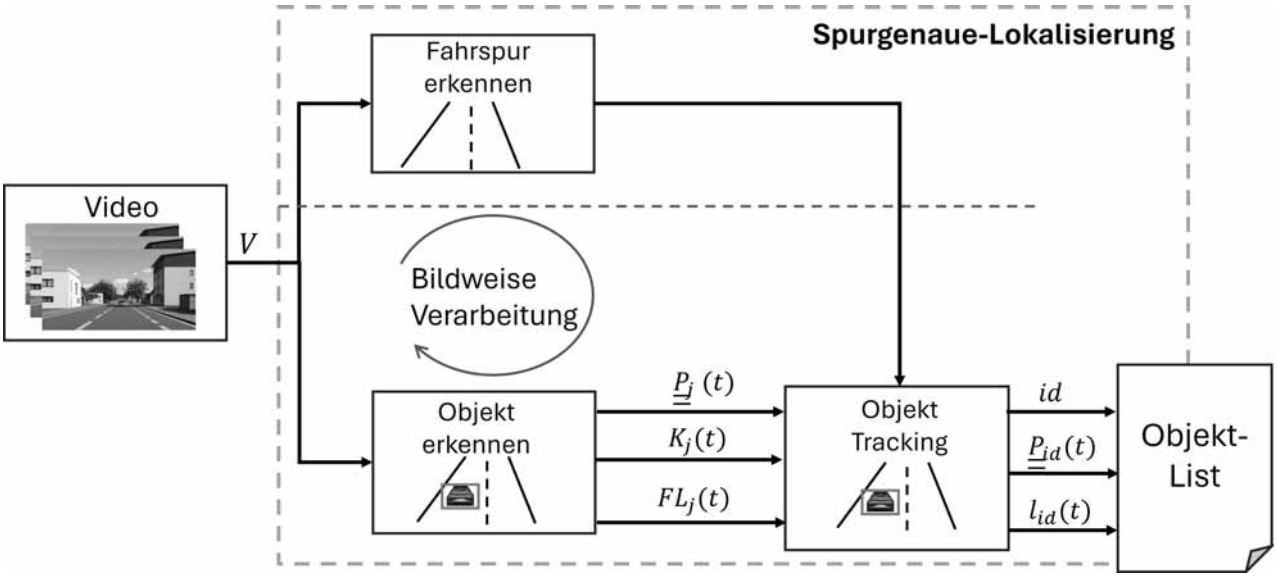

Abbildung 2: Konzeption der spurgenauen Lokalisierung

Prozedur 1 Erkennung der Fahrspur

1: INPUT:V

2: OUTPUT: $l_i(x)$

3: **for** $\underline{\underline{B}}(t)$ in V **do**

4: Kodieren: $\underline{\underline{F}}(t) = BiSeNet(\underline{\underline{B}}(t))$

5: Dekodieren: $\underline{\underline{B}}^b(t), \underline{\underline{B}}^I(t) = Decode(\underline{\underline{F}}(t))$

6: Filter: $\underline{\underline{B}}^f(t) = Filter(\underline{\underline{B}}^b(t))$

7: Cluster: $C_i(t) = DBSCAN(\underline{\underline{B}}^b(t), \underline{\underline{B}}^I(t))$

8: Fahrspur: $l_i(x) = Ausgelichung(C_i(t))$

9: **end for**

ne Effizient der Objekte im Video erfordert, sind die Anforderungen an die Verarbeitungszeit und die Genauigkeit des Erkennungsalgorithmus sehr hoch. Nach einem Vergleich verschiedener Objekterkennungsalgorithmen wurde der YOLO Algorithmus ausgewählt, da er in Bezug auf Geschwindigkeit und Genauigkeit überlegen ist. Nach der Erkennung eines Fahrzeugs erstellt der Algorithmus automatisch eine Box $\underline{\underline{P}}_{j(t)}$ um das Fahrzeug, der durch vier Koordinatenpunkte enthält. j ist die Nummer des erkannten Fahrzeugs. Diese Box befindet sich in der Nähe der umgebenden Kanten des Fahrzeugs, so dass der Mittelpunkt der unteren Seite der Box als Position des Fahrzeugs betrachtet werden kann. Zusätzlich wird im Erkennungsprozess ein Konfidenzwert $K_j(t)$ erzeugt, der die Wahrscheinlichkeit angibt, dass das erkannte Ergebnis zur Fahrzeugkategorie $FL_j(t)$ gehört. Dieser Konfidenzwert bestätigt nicht nur die Erkennungsgenauigkeit, sondern ist auch ein wichtiger Matchingparameter, um das erkannte Fahrzeug dem Objekttracker zuzuführen, damit wird sichergestellt, dass der Tracker die Fahrzeuge genau verfolgen kann.

Nach der erfolgreichen Erkennung von Fahrzeugen wird in dieser Arbeit der Byte-Track-Algorithmus verwendet, um die erkannten Fahrzeuge zu verfolgen. Die Hauptfunktion von Byte-Track besteht darin, jedem erkannten Fahrzeug eine eindeutige Identifikationsnummer zuzuweisen, die nach Erhalt der Position des erkannten Fahrzeugs und der Konfidenzwerte vergeben wird. Anhand dieser ID können die Positionen desselben Fahrzeugs in aufeinander folgenden Videobildern verfolgt. Durch die Analyse der Positionsdaten des Fahrzeugs in Bezug auf die Fahrspur kann außerdem festgestellt werden, dass sich das Fahrzeug auf einer bestimmten Fahrspur befindet. Byte-Track zeichnet auch die Position jedes Fahrzeugs in jedem Bild auf, um den Fahrweg des Fahrzeugs zu rekonstruieren. Diese Funktionen verbessern nicht nur die Kontinuität und Genauigkeit der Verfolgung, sondern liefern auch wichtige Daten über die Dynamische Information der Fahrzeuge. Das Verfahren wird durch Prozedur 2 dargestellt, wobei s angibt, wie vielen Metern jedes Pixel entspricht.

Prozedur 2 Lokalisierung

INPUT: V, $l_i(x)$, s,r

OUTPUT: $t, id, \underline{P}_{id}(t), v_{id}(t), l_{id}(t))$

for $\underline{\underline{B}}(t)$ in V **do**

$f = f + 1$

$\underline{\underline{B}}(t)$ = $addline(\underline{\underline{B}}(t), l_i(x))$

$\underline{P}_j, K_j(t), FL_j(t) = YOLO(\underline{\underline{B}}(t))$

$\underline{P}(t)^{j,B}$, $ID = Track(\underline{P}_j, K_j(t), FL_j(t))$

$\underline{P}_{id}(t) = Box\_Mittelpunktder\_Unterseite(\underline{P}(t)^{j,B}, id)$

if $\underline{P}_{id}(t)[x] > l_i^{-1}(\underline{P}_{id}(t)[y])$ **then**

$l_{id}(t)$ ist i

end if

$Save(t, id, \underline{P}_{id}(t), l_{id}(t))$

end for

5 Ergebnisse

In diesem Kapitel werden die Ergebnisse der Spurgenaue Lokalisierung dargestellt. Diese Funktion wird in einer virtuellen Simulationsumgebung validiert.

Abbildung 3 zeigt die Validierung dieser Objekterkennungsfunktion in einem virtuellen Simulationsumgebung. Die Fahrzeuge in dieser Umgebung werden erkannt. Das Fahrzeug in dem rechteckigen Box stellt das erkannte Fahrzeug dar. Auf dem Kasten befindet sich eine Nummer, die der Identifikationsnummer des erkannten Fahrzeugs bedeutet.

Abbildung 3: Identifizierte Fahrzeuge

Abbildung 4 zeigt die erkannten Fahrspuren in der Simulationsumgebung. Die Linien stellen die angepassten Fahrspuren dar, die mit den tatsächlichen Fahrspuren im Bild übereinstimmen.

Abbildung 4: Erkannte Fahrspuren

Abbildung 5 zeigt die befahrbaren Bereiche der Fahrspuren und ihre Fahrbahnnummern. Die Straße auf der linken Seite ist mit 1 gekennzeichnet, die Straße in der Mitte mit 2, und der Fahrradweg auf der rechten Seite ist nicht erkannt. Zu diesem Zeitpunkt befinden sich sowohl das Ego-Fahrzeug als auch das erkannte Fahrzeug auf Fahrspur 2.

Abbildung 5: Numerierte Fahrspur

6 Zusammenfassung und Ausblick

In der vorliegenden Arbeit wurden sowohl die Selbstlokalisierung auf Fahrspurebene als auch die Lokalisierung anderer Fahrzeuge auf Fahrspurebene entwickelt. Diese Arbeit besteht aus zwei Aspekten: Erstens wird ein Modell auf der Grundlage einer Kodierungs- und Dekodierungsstruktur angewandt, um die Fahrspuren zu identifizieren ununnd zu nummerieren. Zum anderen wird der Algorithmus YoLo verwendet, um andere Fahrzeuge auf der Straße zu identifizieren und deren Fahrspur zu berechnen.Die Funktion wurde in einem

szenariobasierten virtuellen Simulationsumgebung validiert.

Zukünftige Arbeiten werden sich auf die Optimierung der Straßenerkennungsfunktion zur Anpassung an andere Umgebungen konzentrieren. Darüber hinaus werden weitere Sensoren wie LiDAR fusioniert und die erkannten Informationen über V2X-Paare an andere Verkehrsteilnehmer übermittelt werden.

Danksagung

Gefördert vom Niedersächsischen Ministerium für Wissenschaft und Kultur unter Fördernummer ZN4172 im Niedersächsischen Vorab der VolkswagenStiftung.

Literatur

[1] Francini M, Chieffallo L, Palermo A, Viapiana MF. Systematic Literature review on smart mobility: A framework for future “quantitative” developments. *Journal of Planning Literature*. 2021;36(3):283–296.

[2] SAE International. Taxonomy and Definitions for Terms Related to On-Road Motor Vehicle Automated Driving Systems. 2021;SAE Standard J3016_202104.

[3] Rabe J, Necker M, Stiller C. Ego-lane estimation for lane-level navigation in urban scenarios. 2016; pp. 896–901.

[4] Qian C, Zhang H, Li W, Tang J, Liu H, Li B. Cooperative GNSS-RTK ambiguity resolution with GNSS, INS, and LiDAR data for connected vehicles. *Remote Sensing*. 2020;12(6):949.

[5] Ding W, Hou S, Gao H, Wan G, Song S. LiDAR Inertial Odometry Aided Robust LiDAR Localization System in Changing City Scenes. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020; pp. 4322–4328.

[6] Liu-Henke X. *Mechatronische entwicklung der aktiven Feder-, Neigetechnik für das schienenfahrzeug RailCab*. VDI Verlag. 2005.

[7] Li J, Jiang F, Yang J, Kong B, Gogate M, Dashtipour K, Hussain A. Lane-DeepLab: Lane semantic segmentation in automatic driving scenarios for high-definition maps. *Neurocomputing*. 2021; 465:15–25.

[8] Wang Z, Ren W, Qiu Q. Lanenet: Real-time lane detection networks for autonomous driving. *arXiv preprint arXiv:180701726*;.

[9] Wu Z, Qiu K, Yuan T, Chen H. A method to keep autonomous vehicles steadily drive based on lane detection. *International Journal of Advanced Robotic Systems*. 2021;18(2):17298814211002974.

[10] Dewangan DK, Sahu SP, Sairam B, Agrawal A. VLDNet: Vision-based lane region detection network for intelligent vehicle system using semantic segmentation. *Computing*. 2021;103(12):2867–2892.

[11] Lu S, Luo Z, Gao F, Liu M, Chang K, Piao C. A fast and robust lane detection method based on semantic segmentation and optical flow estimation. *Sensors*. 2021;21(2):400.

[12] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:14091556*;.

[13] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. 2015.

[14] Yu C, Gao C, Wang J, Yu G, Shen C, Sang N. BiSeNet V2: Bilateral Network with Guided Aggregation for Real-time Semantic Segmentation. 2020.

[15] Xiangbing LI, Lian C. Face Detection in Natural Scene Based on Improved Faster-RCNN[J]. *Computer Engineering*. 2021;47(1):210–216.

[16] Huang KQ, Ren WQ, Tan TN, et al. A review on image object classification and detection. *Chinese Journal of Computers*. 2014;37(6):1225–1240.

[17] Liu H, Ma J, Zhang G. Review of studies on deep learning-based content recommendation algorithms [J]. *Computer Engineering*. 2021;47(07):1–12.

[18] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features. In: *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, vol. 1. Ieee. 2001; pp. I–I.

[19] Felzenszwalb P, McAllester D, Ramanan D. A discriminatively trained, multiscale, deformable part model. In: *2008 IEEE conference on computer vision and pattern recognition*. Ieee. 2008; pp. 1–8.

[20] Girshick R, Donahue J, Darrell T, Malik J. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014; pp. 580–587.

[21] Girshick R. Fast R-CNN. In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. 2015; .

[22] Cai Z, Vasconcelos N. Cascade R-CNN: Delving Into High Quality Object Detection. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018; .

[23] Lin TY, Dollár P, Girshick R, He K, Hariharan B, Belongie S. Feature pyramid networks for object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017; pp. 2117–2125.

[24] Redmon J, Divvala S, Girshick R, Farhadi A. You only look once: Unified, real-time object detection. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016; pp. 779–788.

[25] Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. In: *2017 IEEE international conference on image processing (ICIP)*. IEEE. 2017; pp. 3645–3649.

[26] Zhang Y, Sun P, Jiang Y, Yu D, Weng F, Yuan Z, Luo P, Liu W, Wang X. Bytetrack: Multi-object tracking by associating every detection box. In: *European conference on computer vision*. Springer. 2022; pp. 1–21.