Spurerkennung aus LiDAR- und Kameradaten mittels Deep Learning

Maximilian Dreyer
Universität Potsdam, Digital Engineering Fakultät, Hasso-Plattner-Institut

Zusammenfassung

Die Erkennung von Fahrbahnspuren ist ein fundamentales Problem beim autonomen Fahren. Die Lokalisation des Fahrzeugs muss präzise und zuverlässig sein, damit Pfad-und Trajektorienplanung funktionieren können. In dieser Seminararbeit wird ein Deep Learning-Ansatz vorgestellt, welcher mithilfe von LiDAR- und Kameradaten Spuren in der Vogelperspektive ermittelt. Die Lokalisation kann anschließend über den Vergleich mit einer leichtgewichtigen zweidimensionalen HD-Map durchgeführt werden. In komplexen städtischen Situation werden Unsicherheiten von 0,18 m im Mittel in der lateralen Lokalisation erreicht.

1 Einleitung

Autonom fahrende Fahrzeuge verfügen meist über eine hochauflösende Karte (HD-Map) ihrer Umgebung, die relevante Informationen wie Fahrspuren, Verkehrszeichen und oft auch Verkehrsregeln enthält. Damit das Fahrzeug mit der Karte seine zukünftige Trajektorie planen kann, muss eine präzise Lokalisierung auf der Karte gewährleistet sein. Die Anforderung hinsichtlich Lokalisierungsgenauigkeit liegt bei wenigen Zentimetern.

Für die Lokalisierung können verschiedene Sensoren wie das Global Positioning System (GPS), die Inertial Measurement Unit (IMU) oder Light Detection and Ranging (LiDAR) eingesetzt werden. Das Global Positioning System bestimmt mithilfe mehrer Satelliten die Position. Es ist ein weitverbreitetes System, erreicht aber nur eine Genauigkeit von mehreren Metern - besonders in Tunneln oder neben hohen Gebäuden. Die Inertial Measurement Unit liefert Informationen über die Bewegung des Fahrzeugs wie Beschleunigung und Winkelgeschwindigkeit. Aus der Dynamik und Position der Vergangenheit lässt sich die aktuelle Position des Fahrzeugs bestimmen. Allerdings kommt es aufgrund von Ungenauigkeiten zu einem Drift nach längerer Zeit. Weitere Ansätze basieren auf der Erfassung der Umgebungswelt (z.B. Kamera oder LiDAR) und einem Vergleich mit der HD-Map. Es können geometrische sowie optische Umgebungsmerkmale verglichen werde, beispielsweise mit LiDAR Punktwolken/Intensität.

Diese Ortserkennungsverfahren erhöhen die Lokalisierungspräzision, verlangen aber oft hohe Rechenressourcen. Besonders viel Speicherplatz wird benötigt, wenn die HD-Map als dreidimensionale Punktwolke vorliegt. Wenn große Areale erfasst werden, skaliert ein zweidimensionaler Ansatz besser. In diesem Projekt wird eine leichtgewichtige 2D-HD-Map verwendet, die Informationen über Fahrbahnspuren als Vektorobjekte (Polygone) enthält.

Für das Projekt werden freiverfügbare Daten von nuScenes [Caesar et al. 2019] verwendet. Es werden 600 Szenen mit jeweils 20 Sekunden Länge des öffentliche Datensatzes ausgewählt. In dem Datensatz sind Szenen aus Boston und Singapur bei verschiedenen Wetter- und Lichtbedingungen (z.B. Regen oder Nacht) gegeben. Jede Szene besteht aus 40 Samples mit LiDAR Punktwolken sowie Kamerabildern.

2 Kontext

Die Betreuung im Rahmen der Seminartätigkeit erfolgte durch das Fachgebiet für Computergrafische Systeme, dessen Forschungsschwerpunkt die Prozessierung, Abbildung und interaktive Visualisierung massiver raumzeitlicher [Oehlke et al. 2015; Buschmann et al. 2015; Buschmann et al. 2014; Maass and Döllner 2006b] sowie abstrakter, hochdimensionaler Daten [Limberger et al. 2017; Limberger et al. 2016; Würfel et al. 2015] ist. Dies beinhaltet neben neuartigen Algorithmen [Richter et al. 2013b; Richter et al. 2013a; Glander et al. 2012], Rendering-Techniken [Semmo et al. 2016a; Pasewaldt et al. 2014; Maass and Döllner 2006a; Döllner et al. 2005] und Interaktions-Metaphern [Semmo et al. 2016b; Scheibel et al. 2016; Semmo and Döllner 2014] auch effiziente Datenstrukturen [Scheibel et al. 2017; Richter et al. 2015] und Systemarchitekturen [Klimke et al. 2014; Trapp et al. 2012; Klimke and Döllner 2010], die anhand von real-weltlicher Datensätze und Anwendungsszenarien [Discher et al. 2016; Trapp et al. 2015; Engel et al. 2012] evaluiert werden.

3 Verwandte Arbeiten

In diesem Abschnitt werden verwandte Arbeiten vorgestellt und diskutiert, die sich mit dem Problem der Spurerkennung befassen. Die verschiedenen Ansätze sind folgende Kategorien eingeteilt:

Kamerabild in der Fahrzeugperspektive Ein intuitiver Ansatz besteht darin, die Spuren direkt im Kamerabild zu erkennen. Traditionelle Computervision-Techniken beruhen auf Kanten- und Eckfiltern sowie einem anschließenden Linienfit [Haloi and Jayagopi 2015]. Die traditionellen Methoden sind instabil, wenn es starke Verdeckungen (Verkehr) oder wechselnde Licht- und Wetterverhältnisse gibt.
Fortschritte in der Entwicklung und Performanz von Convolutional Neural Networks (CNN) haben Deep Learning Modelle als weiteren wichtigen Ansatz etabliert. [Lee et al. 2017] haben ein CNN-Model vorgestellt und dabei verwendet, dass Spuren im 3D-Raum meist parallel sind und gemeinsame Fluchtpunkte haben.

Kamerabild in der Vogelperspektive Für den Vergleich mit einer zweidimensionalen HD-Map ist es sinnvoll, die Spuren in der Vogelperspektive zu erkennen. Es bietet sich an, das Kamerabild mittels Inverse Perspective Mapping (IPM), wie in der Projektarbeit verwendet, zu transformieren. [He et al. 2016a] haben gezeigt, dass auch ein siamesisches Modell, welches beide Perspektiven verwendet, möglich ist. Der Nachteil eines einfachen IPM ist, dass eine konstante Bodenhöhe angenommen wird und somit Ungenauigkeiten in der Ortsauflösung auftreten.

LiDAR Ältere Ansätze klassifizieren einzelne Punkte einer LiDAR Punktewolke mit einem anschließenden Kurvenfit [Hata and Wolf 2014]. Aktuellere Arbeiten verwenden ebenfalls CNNs und eine Punktdarstellung in der Vogelperspektive [Caltagirone et al. 2017]. Hier ist es sinnvoll, LiDAR Intensitäten zu verwenden.

HASSO-PLATTNER-INSTITUT
Fachgebiet Computergrafische Systeme

Seminar Visual Analytics for High Dimensional Data
Sommersemester 2020
Themenstellung und Anleitung: Sören Discher, Dr. Rico Richter und Prof. Dr. Jürgen Döllner
http://www.hpi3d.de

Multi-Sensor Die Verwendung von mehreren Sensoren verbessert die Performanz sowie Stabilität der Spurerkennung. In der Arbeit von [Huang et al. 2009] wird die Kameraaufnahme für die Spurerkennung und LiDAR für Objektmaskierung sowie für Bordsteinerkennung verwendet. In der Arbeit [Bai et al. 2018], an dem sich dieses Projekt orientiert, werden Kamera- und LiDAR-Daten simultan in einem neuronalen Netz als Eingabedaten genutzt. Diese Sensorfusion bietet sich auch für 3D Objekterkennung an [Liang et al. 2019].

4 Datenaufbereitung

Das hier verwendete neuronale Netz nimmt als Eingabe Kamera- sowie LiDAR-Daten in der Vogelperspektive entgegen. Hierbei handelt es sich um Bilddateien mit einer Auflösung von 960x960 Pixeln, die eine reale Fläche von 40x40 m abdecken. Die Ausgabe enthält die Fahrspuren, wobei jedem Pixel die minimale Distanz zur nächstgelegenen Spur zugeordnet wird. Mit diesem Ansatz erreichen auch vorhergesagte Spuren, die leicht versetzt zur Groundtruth sind, eine gute Präzision. Dies ist stabiler, als wenn jeder Pixel eindeutig klassifiziert wird.
Für ein einheitliches Trainieren des Modells, wird die minimale Distanz bis zu einem Schwellenwert (1 m) berechnet und von diesem Wert subtrahiert. Somit hat ein Pixel auf der Spur den maximalen Wert, welcher mit wachsender Entfernung auf Null abfällt.

4.1 LiDAR

Für die LiDAR-Eingabe werden die Punktwolken aus 20 vergangen Samples (sofern vorhanden) zusammengeführt, um den Informationsgrad so hoch wie möglich zu halten. Dabei werden sich ändernde Ausrichtungen des Fahrzeugs berücksichtigt (siehe Abb. 1).
Die Eingabe besteht aus drei Kanälen, wobei die Pixel je nach Kanal den maximalen Intensitätswert, minimale oder maximale Höhe in dem Gridbereich annehmen. Um Ausreißern entgegenzuwirken, werden die Intensitätswerte auf $\pm 3\sigma$ Standardabweichungen und die Höhenwerte auf ± 2 m begrenzt.

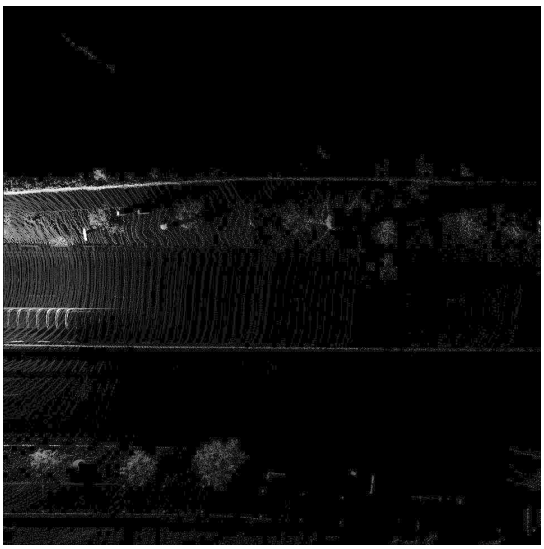

Abbildung 1: *Lidarpunktwolke (Intensität) von 20 vergangen Samples. Spurmarkierungen (gestrichelte Linien) sowie eine Hecke (obere Bildhälfte mit hoher Intensität) sind zu erkennen. Ebenfalls sind die Umrisse eines überholenden Autos sichtbar.*

Die Intensitäten sollen dem neuronalen Netz helfen, die reflektierenden Markierungen zu erkennen. Die Höhen sollen es ermöglichen, verdeckende Objekte wie Autos oder Mauern zu identifizieren.

4.2 Kamera

Das Kamerabild in der Fahrzeugperspektive wird mittels Inverse Perspective Mapping (IPM) in die Vogelperspektive transformiert (siehe Abb. 2 und 3). Das IPM basiert auf einem Lochkameramodell und transformiert 3D Weltkoordinaten $[X, Y, Z]$ in 2D Pixelkoordinaten $[x, y]$. Unter der Verwendung von homogenen Koordinaten ergibt sich [Oliveira et al. 2015]

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \lambda \end{pmatrix} = C \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \tag{1}$$

wobei C eine Transformationsmatrix beschreibt. Die Pixelkoordinaten erhält man durch $x = \hat{x}/\lambda$ sowie $y = \hat{y}/\lambda$.

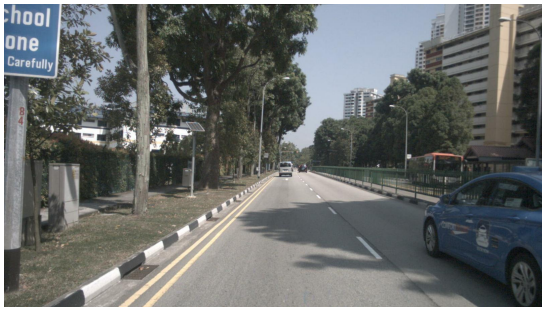

Abbildung 2: *Kamerabild aus der Fahrzeugperspektive. Wie in der LiDAR-Punktwolke zu erkennen, befindet sich eine Hecke auf der linken Seite sowie ein überholendes Auto auf der rechten Seite. Die Spurmarkierungen sind deutlich sichtbar.*

Es wird angenommen, dass die Fahrbahn auf einer konstanten Höhe von Null liegt. Mit $Z = 0$ ergibt sich aus (1) das einfachere Problem einer Homographie

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \lambda \end{pmatrix} = H \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \tag{2}$$

mit der Homographiematrix H. Nun lässt sich für jede Position $[X, Y]$ auf der Fahrbahnebene mittels (2) die Position des Pixels im Ursprungsbild berechnen. Es wird bilineares Sampling verwendet, um das transformierte Bild anschließend zu generieren.

Die Matrix H lässt sich einerseits numerisch aus vier Punkten im Ursprungsbild sowie deren Position im transformierten Bild bestimmen. Da H sich allerdings von Szene zu Szene unterscheidet, ist diese Variante zu aufwändig. Andererseits lässt sich H über die Kalibrierungswerte sowie Position und Ausrichtung der Kamera bestimmen. Diese Daten liegen für jede Szene im Datensatz vor.

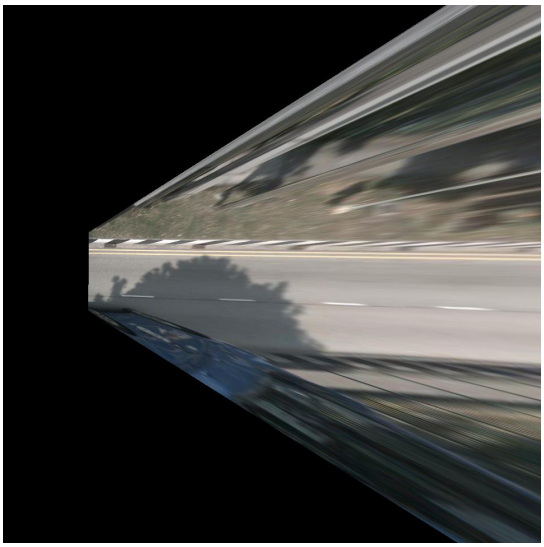

Abbildung 3: *Kamerabild aus der Vogelperspektive erzeugt mittels eines Inverse Perspective Mappings mit Annahme einer konstanten Bodenhöhe. Die Spuren bleiben auch bei größerem Abstand parallel, was auf eine ebene Straße hinweist.*

4.3 Groundtruth

Die Fahrbahnspuren sind in der HD-Map durch Polygon-Segmente gekennzeichnet (siehe Abb. 4). Dabei sind bei Gabelungen oder Kreuzungen meist keine konkreten Fahrbahnspuren vorhanden.

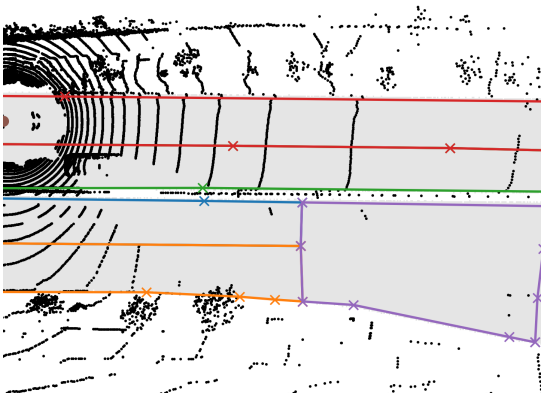

Abbildung 4: *Fahrbahnspuren (grau) der HD-Map aus der Vogelperspektive sowie die aktuelle LiDAR Punktwolke. Gekennzeichnet sind die Polygon-Eckpunkte (farbige Kreuze), die die Fahrbahnen markieren. Da sich die Polygone teilweise überlagern, sind nicht alle Kanten sichtbar. Im Gabelungssegment (rosa) sind keine Fahrbahnmarkierungen enthalten.*

Für die Groundtruth (siehe Abb. 6) soll für jeden Pixel der minimale Abstand zur nächsten Spur ermittelt werden. Damit dies möglich ist, wird folgendes umgesetzt:

1. Aus allen Polygonen werden Punktpaare gebildet, die eine Kante der Spur darstellen.
2. Für jeden Pixel wird die Entfernung zu jedem Punktpaar/jeder Kante berechnet.
3. Der minimale Abstand wird ermittelt und auf den Schwellenwert $d_{\max} = 1\,\text{m}$ begrenzt und von diesem subtrahiert.

Die Entfernung eines Punktes $P = (p_x, p_y)$ von einer Kante zwischen den Punkten $A = (a_x, a_y)$ und $B = (b_x, b_y)$ lässt sich mithilfe des Vektorproduktes berechnen (siehe Abb. 5). Seien $\vec{u} = (p_x - a_x, p_y - a_y)$ und $\vec{w} = (b_x - a_x, b_y - a_y)$, dann beträgt der Abstand d zwischen P und der Geraden durch A und B:

$$d = \frac{|\vec{u} \times \vec{w}|}{|\vec{w}|} = |\vec{u}| \sin \alpha \tag{3}$$

wobei α der Winkel zwischen $\vec{w}$ und $\vec{u}$ ist.

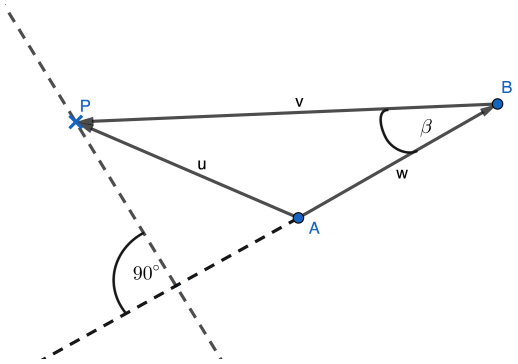

Abbildung 5: *Geometrie zur Bestimmung des Abstandes von Punkt P zur Kante des Punktpaares (A, B). Der minimale Abstand zur Kante ist hier durch u und nicht dem Vektorprodukt aus* (3) *gegeben.*

Allerdings wird aus Abbildung 5 deutlich, dass der Abstand von P zur Geraden (durch A und B) kleiner als der Abstand zur Kante sein kann. Es muss demzufolge noch geprüft werden, ob α sowie β kleiner als $90°$ sind. Ist dies nicht der Fall, ist der minimale Abstand durch $|\vec{u}|$ oder $|\vec{v}|$ gegeben.

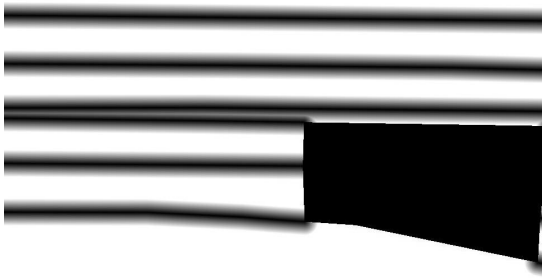

Abbildung 6: *Jeder Pixel der Groundtruth gibt den minimalen Abstand zu einer Fahrspur an (begrenzt auf 1 m). Im Gabelungssegment (schwarze Fläche) sind keine Fahrbahnmarkierungen enthalten.*

Die pixelweise Berechnung der Abstände bei einer Pixelzahl von $960^2 \approx 10^4$ wird vektoriell durchgeführt. Somit wird die Rechenzeit von Minuten auf wenige Sekunden reduziert.

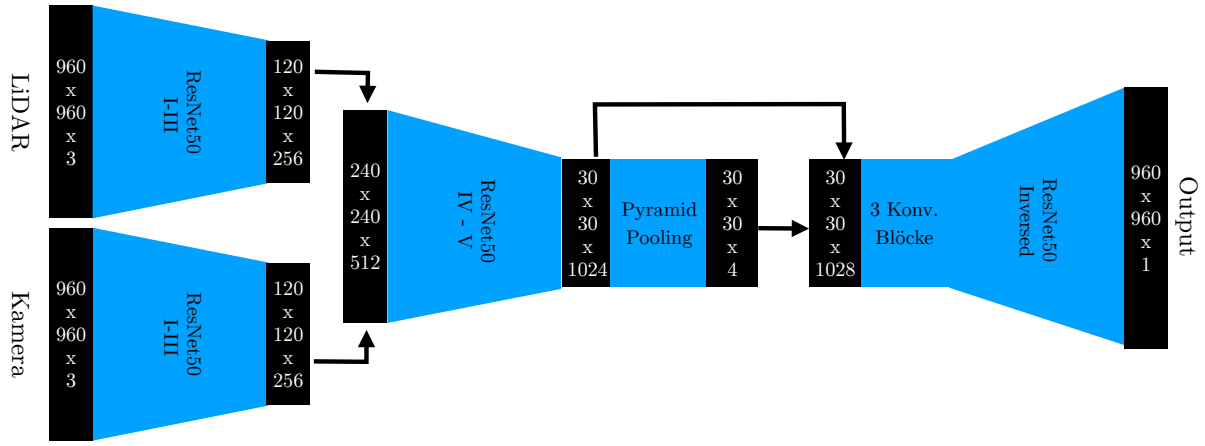

Abbildung 7: *Die Architektur besteht aus zwei getrennten Eingabezweigen, die nach der dritten Schicht des ResNet50-Modells zusammengeführt werden. Anschließend folgen die letzten beiden Schichten sowie ein Pyramid-Pooling-Block. Die Ausgabe dessen sowie des ResNet50-Modells werden in drei weitere Konvolution-Blöcke geführt und letztendlich in einem ResNet-Modell mit transponierten Konvolutionen hochskaliert.*

5 Neuronales Netz - Architektur

Das neuronale Netz beschreibt eine Sanduhr-Architektur und ist an dem ResNet50-Modell [He et al. 2016b] mit halben Filterzahlen angelehnt (siehe Abb. 7).

Im ResNet50-Modell gibt es fünf Abschnitte, wobei jeder Abschnitt die Ausgabedimension halbiert. Die ersten drei Abschnitte werden dupliziert und anschließend zusammengeführt. Somit ergeben sich zwei Eingabezweige, die jeweils getrennt LiDAR- oder Kamerabilder verarbeiten. Nach dem fünften Abschnitt ergibt sich folglich eine Ausgabedimension von 30x30x1024.

Anschließend geht die Ausgabe des modifizierten ResNet-Modells in ein Pyramid Pooling Modul, angelehnt an [Zhao et al. 2017]. Das Modul erzeugt mithilfe von Global Average Pooling Operationen aus der Ausgabe (30x30x1024) vier (AxAx1) Tensoren, wobei $A = \{1, 2, 3, 6\}$. Im weiteren Verlauf werden die Tensoren durch Upsampling auf die Urpsprungsauflösung gebracht und mit der Eingabe zusammengeführt. Damit wird die Information aus großen Teilen der Eingabe zusammengefasst und weiterpropagiert. Dies erhöht das rezeptive Feld.

Der letzte Teil des neuronalen Netzes besteht aus drei weiteren Konvolution-Blöcken sowie einem invertierten ResNet50-Modell. Statt der Konvolutionen mit Stride zwei werden transponierte Konvolutionen mit Stride zwei verwendet.

Die Lossfunktion L, die beim Training minimiert wird, beschreibt einen quadratischen Loss hinsichtlich der Modellparameter θ

$$L(\theta) = \sum_{i=1}^{960} \sum_{j=1}^{960} I_{ij} (p_{ij}(\theta) - g_{ij})^2 \tag{4}$$

mit der Vorhersage p und der Groundtruth g sowie einer Indikatorfunktion I. Die Funktion I ist Null, wenn es für den Pixel keine Spurinformationen gibt (z.B. auf Kreuzungen oder bei Spurgabelungen).

6 Auswertung

Die Datenaufbereitung dauert bei 8 Sekunden Verarbeitungszeit pro Sample und einer Anzahl von 600 Szenen (24000 Samples) ca. 53 Stunden Rechenzeit. Aus über 200 Gb Daten des Datensatzes werden ca. 20 Gb (20.600 Samples) Trainings- sowie 4 Gb (3.400 Samples) Testdaten im HDF5-Format.

Trainieren des Modells Das Training des Machine Learning Modells wird mithilfe des kostenlos zugänglichen Services Google Colaboratory[1] durchgeführt. Dort ist es möglich, Python-Skripte mit Grafikkarten wie Nvidia K80s, T4s, P4s and P100s auszuführen. Trotz der leistungsstarken GPUs ist nur eine Batchsize von zwei möglich. Für das Trainieren einer Epoche wird eine Zeit von zwei bis drei Stunden benötigt.

Während des Trainings wird anhand des quadratischen Losses $L(\theta)$ und der Übereinstimmung $M(\theta)$ abgeschätzt, zu welchem Zeitpunkt die besten Parameter gefunden werden.

$$M(\theta) = \sum_{i=1}^{960} \sum_{j=1}^{960} I_{ij} \left(1 - |p_{ij}(\theta) - g_{ij}|\right) \tag{5}$$

[1] https://research.google.com/colaboratory

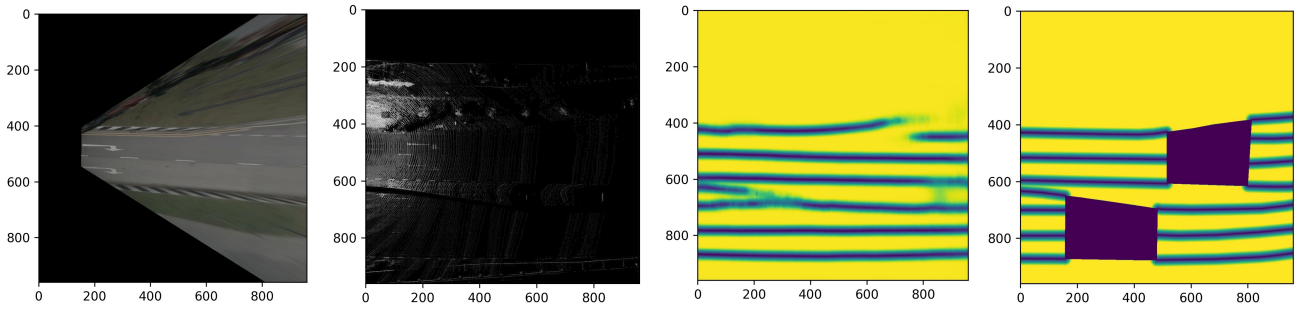

Abbildung 8: *Beispiel für die Vorhersage des neuronalen Netzes auf einer Autobahn mit vielen Spuren (drittes Bild). Mithilfe von Kamera- und LiDAR-Daten (erstes und zweites Bild) können die meisten Spuren der Groundtruth (viertes Bild) vorhergesagt werden.*

Anhand von den Trainingsergebnissen (siehe Abb. 9) ist nach neun Epochen ein Optimum erreicht (Loss: 0,274, Übereinstimmung: 93,12%).

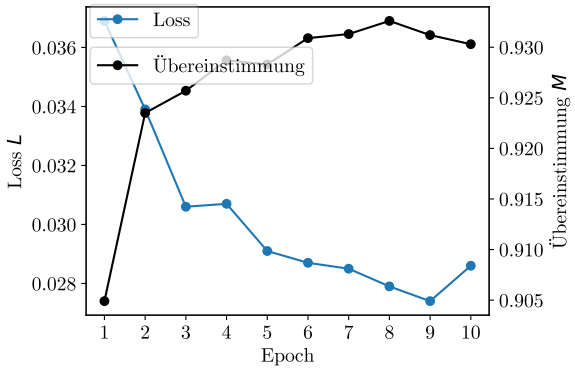

Abbildung 9: *Verlauf des Losses sowie der Übereinstimmung zwischen Vorhersage und Groundtruth für den Testdatensatz. Nach neun Epochen geht der Loss wieder hoch bzw. die Übereinstimmung wird geringer.*

Laterale Lokalisation Um die Performanz des Modells zu überprüfen wird mit den Testdaten eine Lokalisation in lateraler Richtung durchgeführt. Dafür wird die Position der Vorhersage in lateraler Richtung variiert und das Produkt P aus Vorhersage p und Groundtruth g berechnet:

$$P(\Delta) = \sum_{i=1}^{960} \sum_{j=1}^{960} I_{ij} \left(|p_{ij} g_{i,j-\Delta}| \right) \tag{6}$$

mit der Abweichung Δ in Pixeln. Nur wenn Spurvorhersage und Groundtruth überlappen, ist das Produkt ungleich Null. Es wird angenommen, dass die beste Übereinstimmung zwischen g und p das Maximum von P beschreibt. Der laterale Fehler ist somit gegeben durch

$$\epsilon = \Delta_{\text{max}} \frac{40\,\text{m}}{960\,\text{px}} \approx 4{,}16 \Delta_{\text{max}} \frac{\text{cm}}{\text{px}} \tag{7}$$

Im Folgenden wird ϵ auf 5 m begrenzt, da bei einer Sensorfusion davon ausgegangen wird, dass der Fehler durch GPS und IMU wenige Meter beträgt.
Die Positionsfehler in den Samples sind um Null verteilt mit einer Standardabweichung von $\sigma = 0{,}311\,\text{m}$ (siehe Abb. 10). Tatsächlich liegt der mittlere absolute Fehler pro Sample bei $\bar{\epsilon} \approx 18\,\text{cm}$, wenn Fehler über 4 m (0,8% der Samples) nicht betrachtet werden.

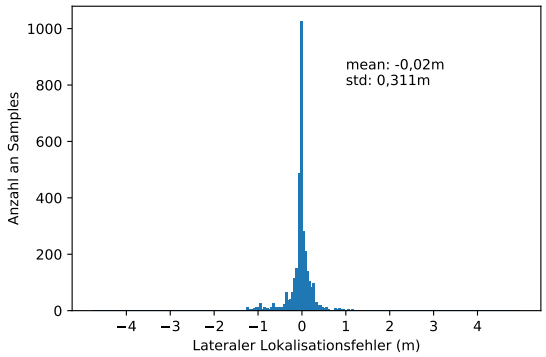

Abbildung 10: *Lateraler Lokalisationsfehler je Sample. Der Großteil der Fehler ist um Null verteilt mit einer Standardabweichung von $\sigma = 0{,}311\,m$.*

Für Samples, wo der Fehler sehr hoch ist, fehlen teilweise Spuren in der Groundtruth oder es gibt keine Spurinformation im Kamerabild oder Groundtruth (siehe Abb. 11). Außerdem ist die Methode der Lokalisation nach (6) nicht optimal. Sind die Spuren der Vorhersage leicht zur Groundtruth verschoben, kann es aufgrund der Multiplikation zu einem geringen P-Wert kommen, obwohl die Spuren optimal übereinanderliegen. Hier wäre ein Ansatz, der die Distanz zwischen den Spuren minimiert (Methode der kleinsten Quadrate), interessant.

Für die Lokalisation werden oft Partikel- oder Kalmanfilter verwendet [Ma et al. 2019] . Mithilfe eines Modells (z.B. Newtonsche Bewegungsgleichungen) und den ermittelten (vergangen und aktuellen) Positionen kann somit eine Lokalisationsposition ermittelt werden, die idealerweise präziser ist. Mithilfe eines Filters kann in diesem Fall die Präzision erhöht werden, denn auch innerhalb der Szenen schwanken die vorhergesagten Positionen um den wahren Wert. Der mittlere laterale Lokalisationsfehler pro Szene liegt bei $\bar{\epsilon}_{\text{Szene}} = -\left(2{,}5 \pm 5{,}1\right)\,\text{cm}$.

7 Fazit

Mithilfe des öffentlich zugänglichen Datensatzes von NuScenes ist es gelungen, Spurerkennung über einen Deep Learning Ansatz durchzuführen (siehe Abb. 8). Der mittlere absolute laterale Fehler liegt hier bei ca. 18 cm. Dies ist vergleichbar mit anderen Ansätzen,

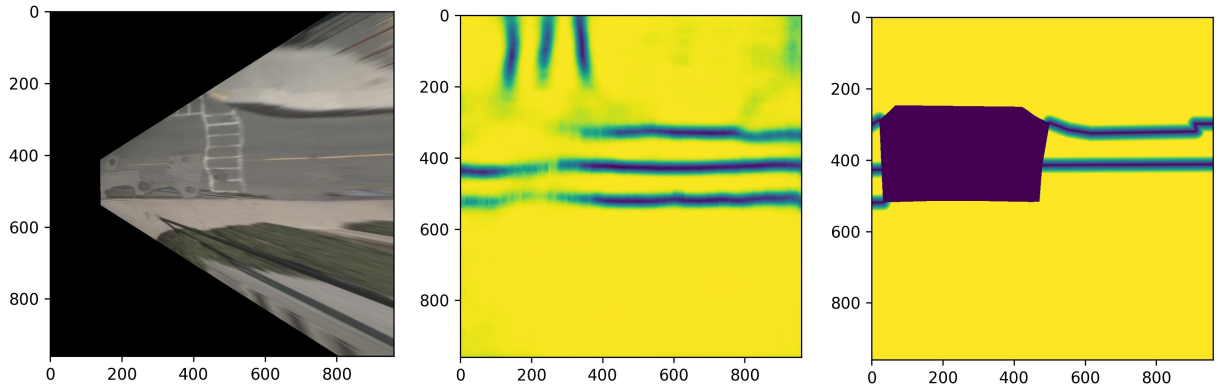

Abbildung 11: *Sample mit einem Lokalisationsfehler von mehr als 4 m. Das Kamerabild (links) zeigt zwei Spuren und deuten eine Kreuzung an, an der Spuren nach oben abzweigen. Die Vorhersage (mitte) zeigt die Abzweigespuren sowie die Hauptspuren korrekt an. In der Groundtruth (rechts) fehlt eine Spurmarkierung und die Spurabzweigung ist ebenfalls nicht vorhanden.*

wie von [Poggenhans et al. 2018] mit einem mittleren Fehler von 14 cm sowie [Ma et al. 2019] mit 20 cm.

Der auf Kamera sowie LiDAR basierende Ansatz ist demzufolge geeignet, die Lokalisation zu unterstützen und die Anforderung von wenigen Zentimetern Lokalisierungsungenauigkeit zu erreichen. [Ma et al. 2019] haben hierfür gezeigt, dass mithilfe von einem ähnlichen Ansatz und Sensorfusion ein mittlerer Fehler von 5 cm möglich ist.

Mit einer Präzision von weniger als 1 m in 96% der Fällen funktioniert die Spurerkennung robust. In den Daten mit einem Fehler von über 1 m fehlen oft Spuren in der Groundtruth. Ferner kann es vorkommen, dass die Spuren durch Verkehr bzw. Baustellen verdeckt sind oder erst gar keine Spurmarkierungen in der realen Welt vorhanden sind. Die Präzision ist auch beschränkt durch den Algorithmus für die laterale Positionsbestimmung. Anstelle eines pixelweisen Vergleichs, wäre ein Ansatz, der die Distanz zwischen allen Spuren minimiert (z.B. über die Methode der kleinsten Quadrate) zu präferieren.

Literatur

BAI, M., MATTYUS, G., HOMAYOUNFAR, N., WANG, S., LAKSHMIKANTH, S. K., AND URTASUN, R. 2018. Deep multi-sensor lane detection. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 3102–3109.

BUSCHMANN, S., TRAPP, M., LÜHNE, P., AND DÖLLNER, J. 2014. Hardware-accelerated attribute mapping for interactive visualization of complex 3d trajectories. In *Proceedings of the 5th International Conference on Information Visualization Theory and Applications*, 355–363.

BUSCHMANN, S., TRAPP, M., AND DÖLLNER, J. 2015. Real-time visualization of massive movement data in digital landscapes. In *Proceedings of the 16th Conference on Digital Landscape Architecture*, 213–220.

CAESAR, H., BANKITI, V., LANG, A. H., VORA, S., LIONG, V. E., XU, Q., KRISHNAN, A., PAN, Y., BALDAN, G., AND BEIJBOM, O. 2019. nuscenes: A multimodal dataset for autonomous driving. *arXiv preprint arXiv:1903.11027*.

CALTAGIRONE, L., SCHEIDEGGER, S., SVENSSON, L., AND WAHDE, M. 2017. Fast lidar-based road detection using fully convolutional neural networks. In *2017 ieee intelligent vehicles symposium (iv)*, IEEE, 1019–1024.

DISCHER, S., RICHTER, R., AND DÖLLNER, J. 2016. Interactive and view-dependent see-through lenses for massive 3d point clouds. In *Advances in 3D Geoinformation*, 49–62.

DÖLLNER, J., BUCHHOLZ, H., NIENHAUS, M., AND KIRSCH, F. 2005. Illustrative visualization of 3d city models. In *Visualization and Data Analysis*, vol. 5669 of *Proceedings of the International Society for Optical Engine*, 42–51.

ENGEL, J., PASEWALDT, S., TRAPP, M., AND DÖLLNER, J. 2012. An immersive visualization system for virtual 3d city models. In *Proceedings of the 20th International Conference on Geoinformatics*, 1–7.

GLANDER, T., TRAPP, M., AND DÖLLNER, J. 2012. Concepts for automatic generalization of virtual 3d landscape models. *gis.SCIENCE 25*, 1, 18–23.

HALOI, M., AND JAYAGOPI, D. B. 2015. A robust lane detection and departure warning system. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 126–131.

HATA, A., AND WOLF, D. 2014. Road marking detection using lidar reflective intensity data and its application to vehicle localization. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 584–589.

HE, B., AI, R., YAN, Y., AND LANG, X. 2016. Accurate and robust lane detection based on dual-view convolutional neutral network. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 1041–1046.

HE, K., ZHANG, X., REN, S., AND SUN, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

HUANG, A. S., MOORE, D., ANTONE, M., OLSON, E., AND TELLER, S. 2009. Finding multiple lanes in urban road networks with vision and lidar. *Autonomous Robots 26*, 2-3, 103–122.

KLIMKE, J., AND DÖLLNER, J. 2010. Combining synchronous and asynchronous collaboration within 3d city models. In *Proceedings of the 6th GIScience*, 115–129.

KLIMKE, J., HAGEDORN, B., AND DÖLLNER, J. 2014. Scalable multi-platform distribution of spatial 3d contents. *International Journal of 3-D Information Modeling 3*, 3, 35–49.

LEE, S., KIM, J., SHIN YOON, J., SHIN, S., BAILO, O., KIM, N., LEE, T.-H., SEOK HONG, H., HAN, S.-H., AND SO KWEON,

I. 2017. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *Proceedings of the IEEE international conference on computer vision*, 1947–1955.

LIANG, M., YANG, B., CHEN, Y., HU, R., AND URTASUN, R. 2019. Multi-task multi-sensor fusion for 3d object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 7345–7353.

LIMBERGER, D., FIEDLER, C., HAHN, S., TRAPP, M., AND DÖLLNER, J. 2016. Evaluation of sketchiness as a visual variable for 2.5d treemaps. In *Proceedings of the 20th International Conference of Information Visualization*, 183–189.

LIMBERGER, D., SCHEIBEL, W., HAHN, S., AND DÖLLNER, J. 2017. Reducing visual complexity in software maps using importance-based aggregation of nodes. In *Proceedings of the 8th International Conference on Information Visualization Theory and Applications*. 176–185.

MA, W.-C., TARTAVULL, I., BÂRSAN, I. A., WANG, S., BAI, M., MATTYUS, G., HOMAYOUNFAR, N., LAKSHMIKANTH, S. K., POKROVSKY, A., AND URTASUN, R. 2019. Exploiting sparse semantic hd maps for self-driving vehicle localization. *arXiv preprint arXiv:1908.03274*.

MAASS, S., AND DÖLLNER, J. 2006. Dynamic annotation of interactive environments using object-integrated billboards. In *Proceedings of the 14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 327–334.

MAASS, S., AND DÖLLNER, J. 2006. Efficient view management for dynamic annotation placement in virtual landscapes. In *Proceedings of the 6th International Symposium on Smart Graphics*, 1–12.

OEHLKE, C., RICHTER, R., AND DÖLLNER, J. 2015. Automatic detection and large-scale visualization of trees for digital landscapes and city models based on 3d point clouds. In *Proceedings of the 16th Conference on Digital Landscape Architecture*, 151–160.

OLIVEIRA, M., SANTOS, V., AND SAPPA, A. D. 2015. Multimodal inverse perspective mapping. *Information Fusion 24*, 108–121.

PASEWALDT, S., SEMMO, A., TRAPP, M., AND DÖLLNER, J. 2014. Multi-perspective 3d panoramas. *International Journal of Geographical Information Science 28*, 10, 2030–2051.

POGGENHANS, F., SALSCHEIDER, N. O., AND STILLER, C. 2018. Precise localization in high-definition road maps for urban regions. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2167–2174.

RICHTER, R., BEHRENS, M., AND DÖLLNER, J. 2013. Object class segmentation of massive 3d point clouds of urban areas using point cloud topology. *International Journal of Remote Sensing 34*, 23, 8408–8424.

RICHTER, R., KYPRIANIDIS, J. E., AND DÖLLNER, J. 2013. Out-of-core gpu-based change detection in massive 3d point clouds. *Transactions in GIS 17*, 5, 724–741.

RICHTER, R., DISCHER, S., AND DÖLLNER, J. 2015. Out-of-core visualization of classified 3d point clouds. In *3D Geoinformation Science: The Selected Papers of the 3D GeoInfo 2014*, 227–242.

SCHEIBEL, W., TRAPP, M., AND DÖLLNER, J. 2016. Interactive revision exploration using small multiples of software maps. In *Proceedings of the 7th International Conference on Information Visualization Theory and Applications*, 131–138.

SCHEIBEL, W., BUSCHMANN, S., TRAPP, M., AND DÖLLNER, J. 2017. *Attributed Vertex Clouds*, 8 ed. GPU Pro. 3–22.

SEMMO, A., AND DÖLLNER, J. 2014. An interaction framework for level-of-abstraction visualization of 3d geovirtual environments. In *Proceedings of the 2nd ACM SIGSPATIAL Workshop on MapInteraction*, 43–49.

SEMMO, A., DÖLLNER, J., AND SCHLEGEL, F. 2016. Becasso: Image stylization by interactive oil paint filtering on mobile devices. In *Proceedings of the ACM SIGGRAPH Appy Hour*, 6.

SEMMO, A., DÜRSCHMID, T., TRAPP, M., KLINGBEIL, M., DÖLLNER, J., AND PASEWALDT, S. 2016. Interactive image filtering with multiple levels-of-control on mobile devices. In *Proceedings of the ACM SIGGRAPH Asia Symposium on Mobile Graphics and Interactive Applications*, 2.

TRAPP, M., SEMMO, A., POKORSKI, R., HERRMANN, C.-D., DÖLLNER, J., EICHHORN, M., AND HEINZELMANN, M. 2012. Colonia 3d - communication of virtual 3d reconstructions in public spaces. *International Journal of Heritage in the Digital Era 1*, 1, 45–74.

TRAPP, M., SEMMO, A., AND DÖLLNER, J. 2015. Interactive rendering and stylization of transportation networks using distance fields. In *Proceedings of the 10th International Conference of Computer Graphics Theory and Applications*, 207–219.

WÜRFEL, H., TRAPP, M., LIMBERGER, D., AND DÖLLNER, J. 2015. Natural phenomena as metaphors for visualization of trend data in interactive software maps. In *Proceedings of Computer Graphics and Visual Computing*, 69–76.

ZHAO, H., SHI, J., QI, X., WANG, X., AND JIA, J. 2017. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2881–2890.