KI-FLEX

Rekonfigurierbare Hardwareplattform zur KI-basierten Sensordatenverarbeitung für das autonome Fahren

Abschlussbericht

Förderkennzeichen:	**16ES1031**
Berichtszeitraum:	**01.09.2019 – 31.08.2023**
Laufzeit des Vorhabens:	**01.09.2019 – 31.08.2023**
Zuwendungsempfänger:	Technische Universität Berlin – Fakultät IV - Elektrotechnik und Informatik – Institut für Telekommunikationssysteme Ernst-Reuter-Platz 7 10587 Berlin
Autor:	Bernd Schäufele bernd.schaeufele@dcaiti.com

Inhaltsverzeichnis

1. Motivation und Zielsetzung

1.1 Aufgabenstellung

1.1.1 *Ausgangslage*

Hochautomatisierte Fahrfunktionen können ein effizienteres Fahren ermöglichen und die Sicherheit und den Komfort für die Fahrzeuginsassen erhöhen. Für sicheres und verlässliche automatisierte Fahrfunktionen, werden hauptsächlich KI-Systeme verwendet. Für die Ausführung der modernsten KI-Software mit neuronalen Netzen (NN) wird Hardware benötigt, die vor Projektbeginn noch großen Platz- und hohen Energiebedarf hatte. Das Projekt KI-FLEX zielte darauf ab, die Hardware durch geringere Abmessungen und niedrigeren Energiebedarf zu verbessern. Die Software im Projekt wurde dabei zielgerichtet für diese neuartige Hardware erstellt. Das Projekt bedient damit den aktuellen Bedarf nach multi-sensorischer Umfelderfassung und robuster Positionsbestimmung, die für VDA-Automatisierungsstufen 4 und 5 benötigt werden. Damit die Projektergebnisse große Relevanz für die Praxis besitzen, ist das Ziel eine preisgünstige hochperformante Plattform mit Leistungsaufnahme unter 10 Watt.

Da diese Zielgröße durch Graphics Processing Units (GPU)-Plattformen und große Field Programmable Gate Arrays (FPGA) nicht realisierbar ist, will das Projekt einen flexibel programmierbaren Multi-Core-Deep-Learning-Beschleuniger als ASIC entwickeln. Dies entspricht dem weltweiten Trend, für KI-Algorithmen spezialisierte Hardware-Beschleuniger statt General Purpose-Central Processing Units (CPU) oder GPUs einzusetzen. Im Gegensatz zum Stand der Technik will KI-FLEX ein zukunftsfähiges ASIC entwickeln, das auch nach Integration ins Fahrzeug zum Integrationszeitpunkt noch unbekannte, zukünftige Konzepte und Algorithmen für Neuronale Netze ausführen kann.

1.1.2 *Allgemeine Ziele*

Das Projekt KI-FLEX hatte das Ziel, die Realisierung vollautomatisierter und autonomer Fahrzeuge zu ermöglichen, indem es ein neuartiges ASIC zur Erfassung von Umgebungsinformationen und für die präzise Eigenlokalisierung entwickelte. Um die Einsatzfähigkeit zu demonstrieren, wurden Verfahren wie der Abgleich von Umfeldmessdaten mit einer High Definition (HD)-Karte für die Lokalisierung KI-Algorithmen zur Objekterkennung eingesetzt. Das Projekt strebte an, durch den Einsatz von neuartigen NN in den Algorithmen für LIDAR- und Kamera-Sensoren gleichbleibende oder höhere Genauigkeit mit geringerem Energieaufwand zu erzielen. Das Projekt zielte darauf ab, die Stabilität, Zuverlässigkeit und Genauigkeit der Algorithmen zu verbessern. Dabei wurden auch NNs für die Klassifizierung von LIDAR-Punktwolken und fusionierten Rohdaten mehrerer Sensoren eingesetzt. Die entwickelten Algorithmen und Prozessoren zielen darauf ab, in der Serienentwicklung integrierbar, zuverlässig und sicher zu sein.

1.1.3 *Aufgaben*

- Umsetzung der statischen Umgebungserkennung.
- Umsetzung der dynamischen Umgebungserkennung
- Entwicklung tiefer neuronaler Netze (DNN), um die semantische Erkennung von statischer und dynamischer Umgebung zu ermöglichen
- Erkennung der statischen Umgebung für Eigenlokalisierung
- Erweiterter Ansatz des Synchronous Localization and Mapping (SLAM) durch die Verwendung von semantischen Labels
- Leitung des AP 3.3, in dem die Gesamtintegration von Hardware und Software erfolgt.

1.2 Voraussetzungen, unter denen das Vorhaben durchgeführt wurde

Neben der eigentlichen Projektidee war die Zusammensetzung des Konsortiums die entscheidende Voraussetzung für die Durchführung des Projektes.

Die Auswahl der Partner musste die verschiedenen Bereiche für die Entwicklung einer neuartigen Architektur für KI im Bereich automatisierter Fahrzeuge abdecke. Für die KI-Flex-Hardware mussten Partner mit entsprechender Expertise im Bereich des Entwurfs digitaler Schaltungen und digitaler Signalverarbeitung eingebunden werden. Wichtig war auch Know-how auf dem Gebiet der scannenden Laser-Messtechnik einzubinden. Von besonderer Bedeutung waren auch Beteiligte der Bereiche des Maschinellen Lernens und der eingebetteten Systeme. Ebenfalls sollten Experten mit Erfahrung im Entwurf und in der Integration von Hardware und Systemsoftware im Projekt vertreten sein. Die Zusammensetzung der beteiligten Stakeholder sollte sowohl Industrieunternehmen als auch Forschungseinrichtungen und Hochschulen beinhalten.

Aus diesen Anforderungen fanden sich auf Basis eines Konsortialvertrages folgende Partner zusammen:

- Ibeo Automotive Systems GmbH, Hamburg
- Infineon Technologies AG, Neubiberg
- Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen
- Technische Universität München
- videantis GmbH, Hannover
- Fraunhofer FOKUS, Berlin
- DCAITI, Technische Universität Berlin

2. Planung und Ablauf des Vorhabens

2.1 Projektstruktur

In Abbildung 1 ist die Projektstruktur dargestellt. Ausgehend von den AP 1.1 und 1.2 wurden die Anforderungen abgeleitet, die in den AP 2.1, 2.2, 2.4 und 2.5 zur Definition der Hardware und Software und schließlich zu deren Entwicklung führte. In AP3 wurden die Komponenten integriert und als Gesamtsystem zusammengeführt. Abschließend wurde die Demonstration einschließlich Tests durchgeführt.

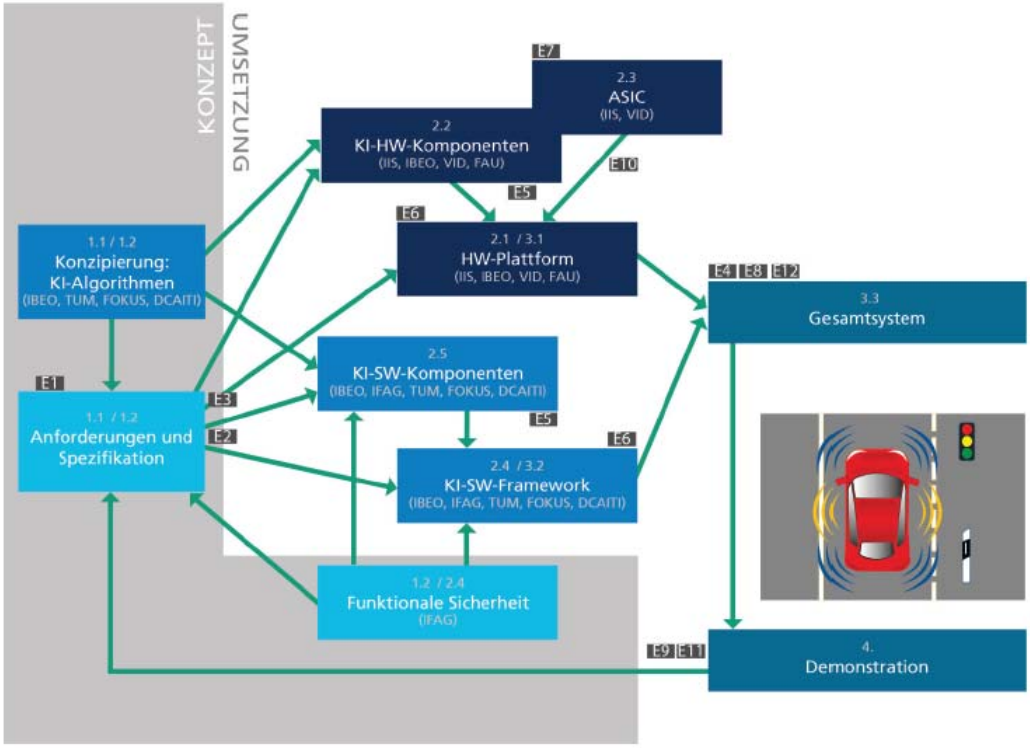

Abbildung 1: Projektstruktur

2.2 Zeitplan und Ablauf

2.2.1 *Zeitplan*

Der ursprüngliche Zeitplan ist in Abbildung 2 dargestellt. Das AP1 legt dabei die Grundlage für die Spezifikation und Entwicklung von AP2. Während AP2 noch lief, war die Integration und Tests vorgesehen.

2.2.2 *Ablauf*

Der Projektablauf konnte letztendlich sichergestellt werden, auch wenn es zu verschiedenen Verzögerungen während der Projektlaufzeit mittelbar und unmittelbar durch die Corona-Pandemie gekommen ist. Zu Verzögerungen in der Bearbeitung und Fertigstellung von Arbeitspaketen für die TU Berlin (DCAITI) waren ursächlich die Verzögerungen bei der Beschaffung von Hardware-Komponenten aufgrund von Störungen in den Lieferketten (z.B. bei Distributoren), Möglichkeiten zur Durchführung von Testfahrten mit Versuchsfahrzeugen waren eingeschränkt, sowie

verzögerte Mitarbeiteranstellungen. Die in beschriebenen Verzögerungen konnten im Rahmen der ursprünglich vorgesehenen Projektlaufzeit nicht ausgeglichen werden. Im Rahmen des Konsortiums wurde daher ein Antrag auf Projektverlängerung gestellt.

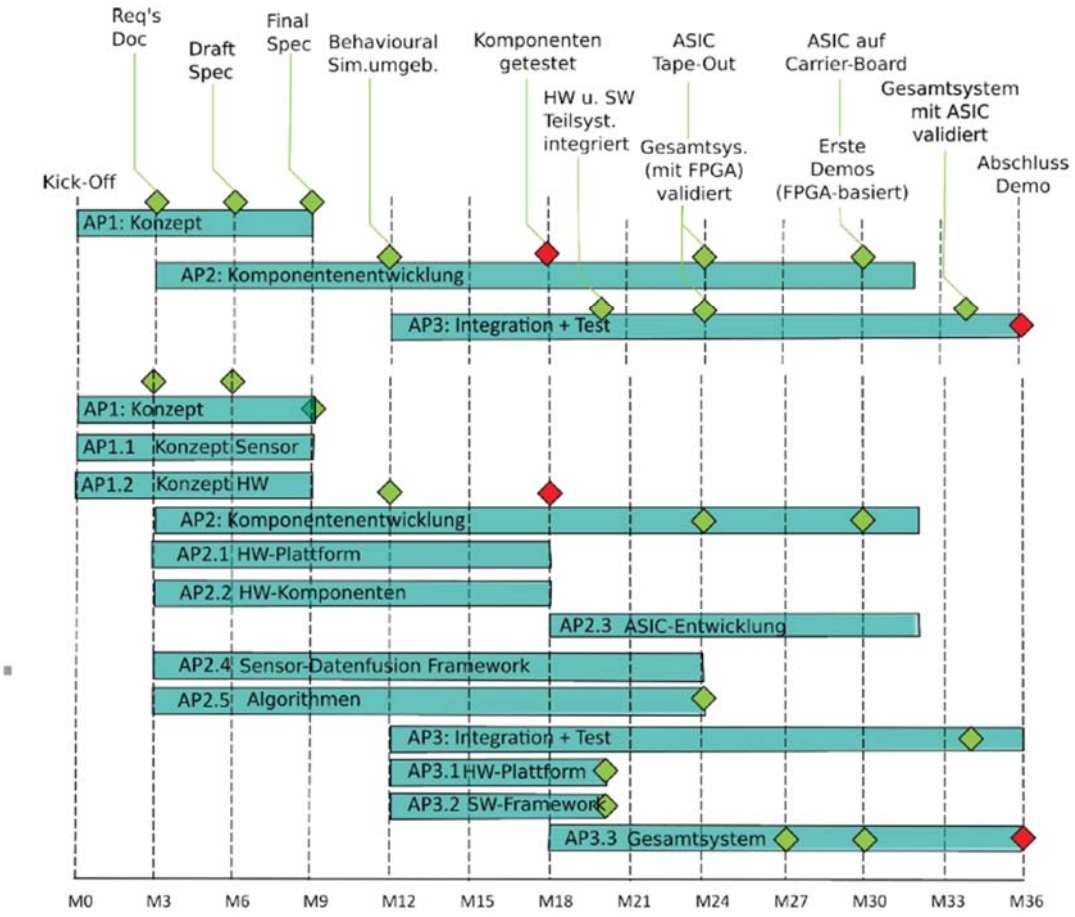

Abbildung 2: Zeitplan

3. Eingehende Darstellung

3.1 Verwendung der Zuwendung und erzielte Ergebnisse

3.1.1 *AP1 Anforderungen und Konzepte*

In AP 1.2.5. wurden aktuelle Verfahren zur semantischen Analyse von Rohdaten zur Objekterkennung vom DCAITI recherchiert und bewertet. Zur Erkennung von statischen und dynamischen Verkehrsteilnehmern, im Fahrbahnbereich kommen Deep Learning Verfahren zum Einsatz. Deep Learning Verfahren werden mit Hilfe von speziellen neuronalen Netzen (engl. Convolutional Neural Networks (CNNs)) implementiert. Dank einer besonderen Netzarchitektur, die eine 3D-Anordnung von Neuronen im Netz ermöglicht, eignen sich CNNs für die Verarbeitung von Bilddaten. Abhängig vom Detaillierungsgrad des gewünschten Ergebnisses wird bei der Erkennung zwischen Klassifikation, Lokalisation und Detektion von Objekten

unterschieden. Bei einer Klassifikationsaufgabe findet die Zuweisung von Bildern zu vordefinierten Objektkategorien statt. Während der Lokalisation wird zusätzlich zur Objektkategorisierung die Position des Objektes markiert. Die Detektion sieht die Bestimmung der Position aller im Vorfeld vordefinierten Objekte vor.

Für die Beurteilung wurde eine Implementierung mit Visualisierung erstellt. Ein Screenshot der Implementierung kann in der Abbildung 3 gesehen werden. Es handelt sich dabei um das Verfahren Mask R-CNN.

Abbildung 3: Beispiel für semantische Segmentierung mit Mask R-CNN

In **AP1.2.8** hat das DCAITI ein Kartenformat entwickelt, mit dem die von anderen Projektpartnern zu entwickelnden Algorithmen arbeiten konnten. Das Kartenformat ist generisch, so dass es mit Karten von verschiedenen Kartenherstellern befüllt werden kann. Es handelt sich um ein textbasiertes Dateiformat, das verschiedene Kartenelemente enthalten kann. In KI-FLEX wurden die Kartendaten zur Lokalisierung verwendet. Dafür werden „Poles“, d.h. längliche vertikale Objekte im Straßenraum verwendet, wie z.B. Straßenlaternen, Ampelmasten oder Pfosten für Verkehrsschilder. Die Dateien sind im ASCII-Format gespeichert worden. Es handelt sich um Koordinaten, bei denen die einzelnen Werte (UTM Northing, UTM Easting, UTM-Zone) per Leerzeichen getrennt sind. Pro Zeile ist ein Elementeintrag enthalten.

3.1.2 *AP2 Komponentenentwicklung*

In **AP2.4.2** wurden das Subsystem zur Umfeldwahrnehmung Requirements erarbeitet. Dazu gehören die Use Cases, die wie folgt definiert wurden.

- UC_KI-FLEX_1: Automatisiertes Fahren im Stadtverkehr
  - Alternativszenario: Stehendes Fahrzeug
- UC_KI-FLEX_2: Innerstädtischer Verkehr statische Objekte

    - Erstes Szenario: Stehendes Fahrzeug
    - Zweites Szenario: Fahrendes Fahrzeug
    - Alternativszenario: Kontrollierte Umgebung (z.B. privates Gelände)
- UC_KI-FLEX_3: Innerstädtischer Verkehr dynamische Objekte
    - Erstes Szenario: Stehendes Fahrzeug
    - Zweites Szenario: Fahrendes Fahrzeug
    - Alternativszenario: Kontrollierte Umgebung (z.B. privates Gelände)

Darüber hinaus wurde eine Reihe von Demonstrationsszenarien definiert. Diese unterteilen sich in das Hauptszenario und Demonstrationsszenarien im Stand

Hauptszenario

- Das Fahrzeug ist mit dem vorgesehenen Hardware-Aufbau einschließlich der entsprechenden Sensoren ausgestattet
- Alle Sensoren sind voll funktionsfähig
- Alle Berechnungen innerhalb der KI-FLEX-Plattform werden an Bord des Fahrzeugs auf der dafür vorgesehenen Hardware live ausgeführt
- Das Fahrzeug bewegt sich durch den Straßenverkehr in städtischer Umgebung
- Für die befahrene Route ist eine zur Lokalisierung geeignete Karte vorhanden
- Das Fahrzeug wird manuell gefahren, eine automatisierte Fahrfunktion ist nicht vorhanden oder deaktiviert

Demonstrationsszenarien im Stand

- Das Fahrzeug ist mit dem vorgesehenen Hardware-Aufbau einschließlich der entsprechenden Sensoren ausgestattet
- Alle Sensoren sind voll funktionsfähig
- Die Berechnungen für die Objekte innerhalb der KI-FLEX-Plattform werden an Bord des Fahrzeugs auf der dafür vorgesehenen Hardware live ausgeführt
- Das Fahrzeug befindet sich stehend in einer bekannten Umgebung mit Elementen einer städtischen Umgebung, z.B. Fußgänger oder Verkehrsschilder
- Für die Lokalisierung ist eine geeignete Karte für die unmittelbare Umgebung des Fahrzeugs vorhanden.

Aus diesen Szenarien wurden Anforderungen an den Fahrzeugaufbau definiert. Der Versuchsträger sollte ein Fahrzeug sein, welches in der Lage ist, automatisierte Fahrmanöver durchzuführen. Dafür sollten sowohl die Längsführung als auch die Querführung durch Aktorik steuerbar sein. Für ein Versuchsfahrzeug ist dabei ein Stellmotor für die Lenksäule sowie eine mechanische Betätigung der Gas- und Bremspedale geeignet.

Das Versuchsfahrzeug sollte mit der für die Detektionsalgorithmen vorgesehenen Sensorhardware ausgerüstet sein. Das Fahrzeug sollte über LIDAR-, Kamera- und möglicherweise über RADAR-Sensoren verfügen. Außerdem sollte ein GPS-

Empfänger vorhanden sein und Zugriffe auf den CAN-Bus möglich sein. Die Daten von diesen Sensoren sollten vom KI-FLEX System verarbeitet werden. In einer Visualisierung sollten die erkannten Objekte sowie weitere Informationen, wie z.B. Freiflächen oder Karteninformationen dargestellt werden.

Zur Bestimmung der relativen Position der erkannten Objekte sollten die 3D-Punkte des LIDARs in die 2D-Boxen des Kamerabildes projiziert werden. Dies erfolgte mit Hilfe einer intrinsischen Kameramatrix, die die Abbildung der realen Koordinaten (in m) im Kamerakoordinatensystem zum Bildkoordinatensystem (in Pixel) definiert. Eine extrinsische Matrix definiert die Abbildung des LIDAR-Koordinatensystems zum Koordinatensystem der jeweiligen Kamera des Sensor-Setups. Diese Matrizen müssen für die verwendet Rig-Konfiguration speziell kalibriert werden. Nach Anwendung der entsprechenden affinen Transformation der extrinsischen Matrix und der Projektion mittels homogener Koordinaten durch die intrinsische Matrix, sollte es möglich sein, den 2D-Box-Detektionen der Objekte 3D-Punktmengen des LIDAR zuzuordnen. Dadurch sollte eine Bestimmung der Objektposition in LIDAR-Koordinaten (in m relativ zum LIDAR-Ursprung) ermöglicht werden.

Abbildung 4: Beispiel Detektion mit LIDAR und Kamera

Bei der Erkennung mit LIDAR und Kamera, erhält man die in Abbildung 4 gezeigten Teilergebnisse. In den Grafiken 1, 4 und 5 in Abbildung 4 sieht man die Kamera-Bilder, die frontal und seitlich links und rechts vom Fahrzeug aus aufgenommen werden. Grafik 6 zeigt die Point Cloud mit dem berechneten Freiraum aus dem LIDAR. Grafik 2 zeigt das Occupancy Grid, was daraus erstellt wird und das in KI-FLEX als Input in die Neuronalen Netze für LIDAR genutzt wurde. Daraus wurden die in den Neuronalen Netze die Bounding Boxes in Grafik 3 ermittelt. Letztendlich

wurde die Position der Bounding Boxes wieder in die Kamera-Bilder (1, 4 und 5) zurückgerechnet (rote Rechtecke).

Die Entwicklung des Frameworks zum Kartenzugriff, welches in AP1.2.8 definiert worden ist, wurde im in **AP2.4.6** umgesetzt. Die Kartendaten liegen zunächst in einem ASCII-Format vor, welches in AP1.2.8. definiert wurde. Über Nachrichten, die mit Google Protobuf implementiert worden sind, können alle in den Karten enthaltenen Objekten ausgetauscht werden. Dazu werden ASCII-Daten eingelesen und über die Protobuf-Schnittstelle versendet.

Die Entwicklung der KI-basierten Umfeldwahrnehmung wurde in **AP2.5.1** umgesetzt. Dafür wurde zunächst das Framework aufgesetzt, das zum Trainieren der Neuronalen Netze dienen soll, die in KI-FLEX eingesetzt werden. Für die Erkennung von Objekten wird TensorFlow eingesetzt. Die in KI-FLEX entwickelte Hardware unterstützt ebenfalls die benötigten Aufrufe der TensorFlow API, so dass die neuronalen Netze später problemlos ausgeführt werden können. In KI-FLEX wird vom DCAITI sowohl ein Netzwerk zur Erkennung in Bildern als auch in LIDAR-Daten entwickelt.

Für die Bilderkennung wird ein Single-Shot-Detector eingesetzt und Inception V2 als Feature Extractor genutzt. Die Netzwerkarchitektur für die Erkennung in Bildern kann in Abbildung 5 gesehen werden. Zunächst werden die Bilder auf eine einheitliche Größe verkleinert, die auch von der KI-FLEX Hardware verarbeitet werden kann.

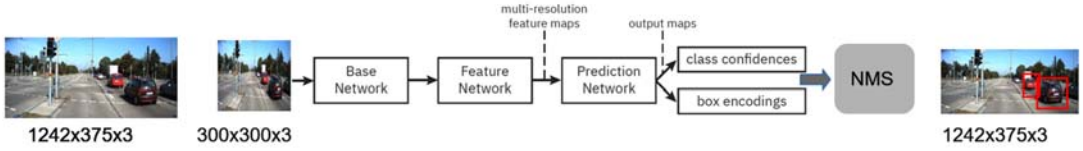

Abbildung 5: Netzwerk-Architektur für Bilderkennung

Dabei wird das Base Network eingesetzt, um grundlegende Merkmale (Basic Features) im Bild zu erkennen. Als nächstes werden im Feature Network mehrere Features zusammengeführt. Dabei werden größere Strukturen erkannt, die bei der Detektion unterstützen. Anschließend wird das Prediction Network ausgeführt, welches hauptsächlich aus Convolutional Layern besteht. Als Ergebnis werden zwei Informationen ausgegeben. Einerseits wird eine Klassifizierung durchgeführt, wobei für die verschiedenen Klassen Konfidenzen ausgegeben werden. Außerdem werden die Objekte durch Bounding Boxes im Bild markiert.

Danach werden die Postprocessing Schritte durchgeführt. Zunächst werden mit der Non maximum Suppression Methode überlappende Bounding Boxes entfernt. Anschließend werden die Daten wieder auf die ursprüngliche Pixelgröße projiziert.

Als letzter Schritt werden mit einer Transformation unter zur Hilfenahme der intrinsischen und extrinsischen Kamera-Kalibrationsmatrix die Pixelkoordinaten in Koordinaten in der realen Welt umgerechnet.

Der zweite Ansatz, der in AP 2.5.1 verfolgt wird, ist die Objekterkennung auf reinen LIDAR-Daten. Dazu werden die LIDAR-Daten zunächst so aufbereitet, dass sie gut mit neuronalen Netzen verarbeitet werden können. Da viele Ansätze zur Erkennung mit neuronalen Netzen auf Bilddaten arbeiten, ist es naheliegend, eine Datenrepräsentierung zu wählen, die Bildern ähnlich ist.

Zu diesem Zweck werden die vom LIDAR-Sensor detektieren Punkte in einzelnen Zellen mit einer Größe von 15 cm gesammelt. Anschließend wird zu jeder Zelle ausgewertet, auf welcher Höhe sich der höchste und der niedrigste Punkt pro Zelle befinden. Zusätzlich wird die maximale Reflektion aus den Punkten einer Zelle bestimmt. Diese Zellen werden auf einer Fläche von 150m x 150m berechnet. Durch die Auflösung von 15cm entsteht somit ein Gitternetz von 1000x1000 Zellen, welches Occupancy Grid genannt wird, da zwischen belegten und unbelegten Zellen unterschieden werden kann. Mit den drei Werten maximale Höhe, minimale Höhe, maximale Reflektivität und 1000x1000 Werten kann ein Drei-Kanal-Bild erstellt werden, bei dem die drei Werte auf die RGB-Kanäle des Bildes gelegt werden. Eine Visualisierung des Occupancy Grids kann in Abbildung 6 gesehen werden. Die farbigen Punkte stellen dabei die belegten Zellen dar.

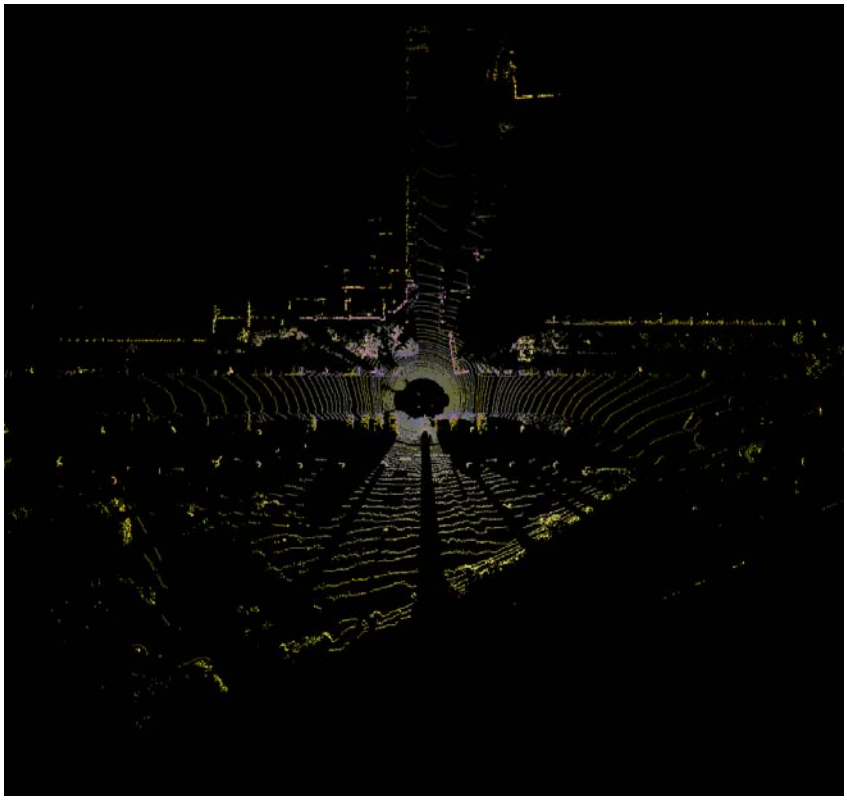

Abbildung 6: Occupancy Grid Darstellung

Für das Training der Netze wurde ein ähnlicher Ansatz wie für die Bild-Daten gewählt. Zunächst wird in einem eigenen Modul im Pre-Processing die Occupancy-Grid-Darstellung berechnet. Durch die drei Netzwerk-Schritte Base Network, Feature Network und Prediction Network werden erneut einfache und zusammengesetzte Features berechnet und abschließend Objektklassen und Bounding Boxes ausgegeben. Da es bei 3D-LIDAR-Daten keine überlappendenden Objekte geben kann, ist kein Post Processing notwendig.

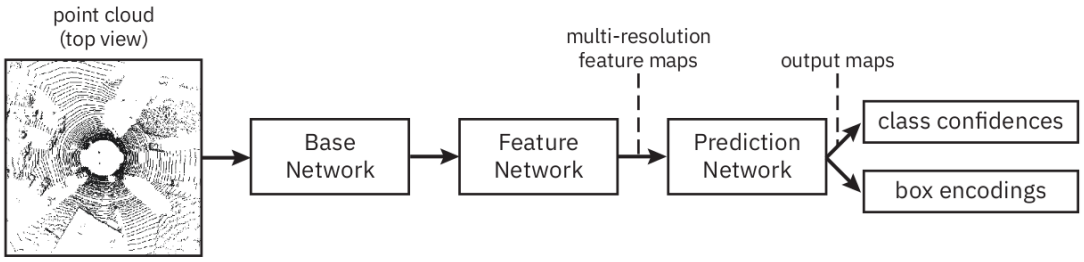

Abbildung 7: Netzwerk-Architektur für LIDAR-Erkennung

In **AP 2.5.5** wurden die Algorithmen zur semantischen Markierung (Labelling) von Bilddaten entwickelt. Dazu wurde zunächst eine Netzwerkarchitektur entwickelt, die auf dem Mask-Scoring RCNN Algorithmus basiert. Ein Screenshot der Erkennung kann in Abbildung 8 gesehen werden.

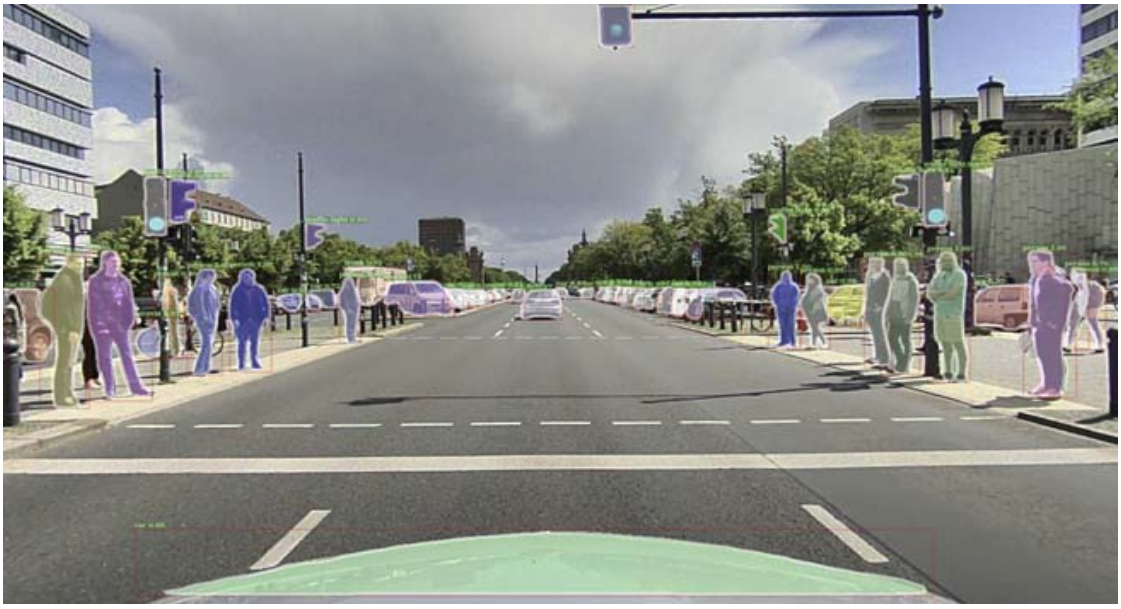

Abbildung 8: Semantische Erkennung

Ein wesentliches Ergebnis war das Training der Netzwerke mit Datensätzen, die vor allem stangenartige Objekte enthalten. Dazu wurden verschiedene Datensätze, die frei verfügbar sind, verwendet.

Für den semantischen SLAM wurde in **AP2.5.7** eine Erkennung entwickelt, die auf einem Partikelfilter basiert. Der Partikelfilter nimmt dazu als Eingangswerte die stangenartigen Objekte aus der Semantischen Erkennung (AP 2.5.5) sowie die Karten aus der Kartenschnittstelle (AP2.4.6). Dazu werden Partikel auf der Karte verteilt und anhand der Abstände zu den Objekten aus der Semantischen Erkennung die Probabilität für die einzelnen Partikel berechnet. Für diese Berechnung werden die Abstände mit den Abständen zu in der Karte enthaltenen Objekten verglichen. Um die Rechenkomplexität zu begrenzen, werden die Partikel zu Beginn nicht auf der ganzen Karte gleich verteilt, sondern es wird anhand des GPS-Signals eine Zone ermittelt, in der die initialen Partikel angelegt werden. Da sich das Fahrzeug zwischen zwei Berechnungs-Schritten bewegt, können die Partikel nicht an derselben Stelle verteilt werden wie bei der Initialisierung. Daher werden vom CAN-Bus des Fahrzeugs auch die Odometrie, d.h. die Geschwindigkeit und der Lenkwinkel, eingelesen, so dass die Eigenbewegung berechnet werden kann und daraus eine ungefähre Schätzung der neuen Position abgeleitet werden kann. Die neuen Partikel werden dann am errechneten ungefähren Aufenthaltsort des Fahrzeugs verteilt. Der Partikelfilter kann dann wiederum durch die Abstandsvergleiche Probabilitäten zuordnen. Ab einem Schrankenwert der Wahrscheinlichkeit wird ein Partikel als Aufenthaltsort des Fahrzeugs akzeptiert.

Für die Erkennung von Poles wurde eine Visualisierung entwickelt. Darin werden die Bildpunkte semantisch eingefärbt und für erkannte Poles die entsprechenden Bounding Boxes eingezeichnet. Die Visualisierung kann in Abbildung 9 gesehen werden.

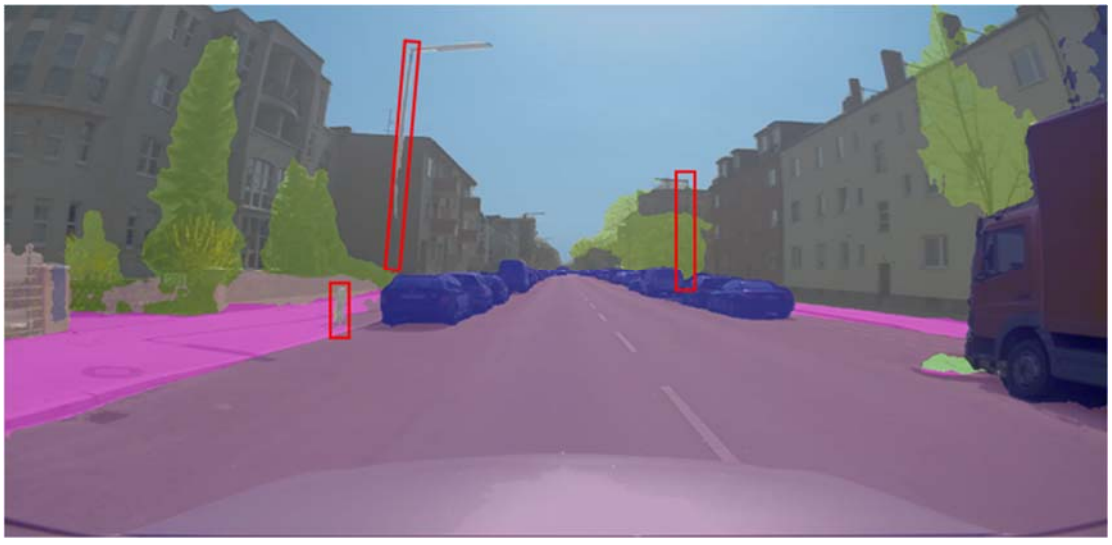

Abbildung 9: Ergebnisse semantische Segmentierung

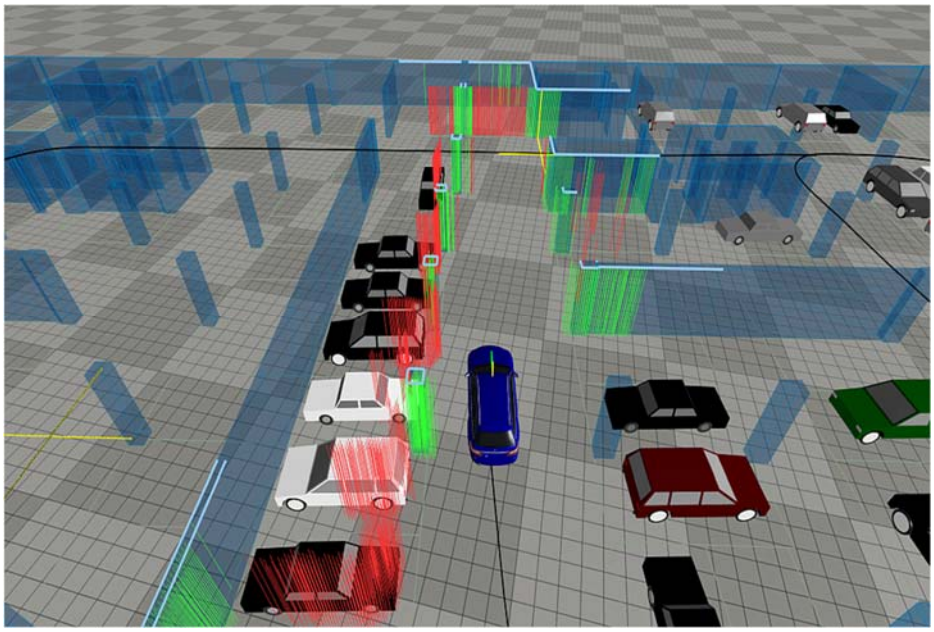

Abbildung 10: Visualisierung der Lokalisierung

Ebenso wurde für die Lokalisierung eine Visualisierung implementiert. Hierfür wurde die Simulationsumgebung PHABMACS verwendet. Im obigen Screenshot kann eine Testfahrt in einer Tiefgarage gesehen werden, bei der Poles und andere erkannte Objekte eingezeichnet werden. Anhand der Poles findet dann die exakte Lokalisierung im 3D-Raum statt.

In **AP 2.5.13** wurden verfügbare Datensätze gesichtet und für KI-FLEX angepasst. Insbesondere der KITTI-Datensatz[1] hat sich dafür als geeignet herausgestellt, da er sowohl gelabelte Bild- als auch LIDAR-Daten enthält. Da die Datenrate auf dem FlexAISIC begrenzt ist, werden die Bild-Daten komprimiert, bevor die Erkennung auf der Hardware durchgeführt wird. Dementsprechend muss das Training der Daten ebenfalls auf komprimierten Daten durchgeführt werden. Dazu wurden verschiedene Untersuchungen vorgenommen mit den Komprimierungsalgorithmen PNG (Komprimierungslevel 3 und 9) und JPG (Qualitätsstufe 95).

- Bildkomprimierung und Evaluation
  - ➔ Training – PNG 3 | Evaluation – PNG 3
  - ➔ Training – PNG 3 | Evaluation – PNG 9
  - ➔ Training – PNG 3 | Evaluation – JPG 95
  - ➔ Training – JPG 95 | Evaluation – JPG 95

[1] http://www.cvlibs.net/datasets/kitti/

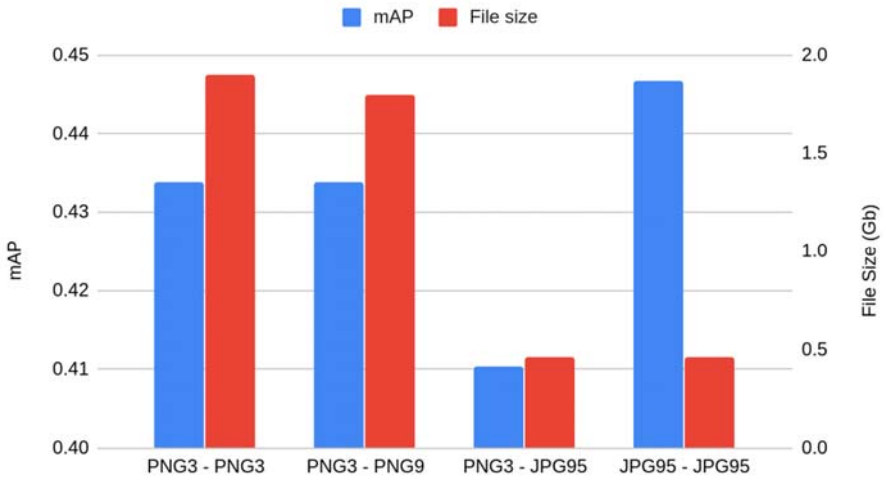

Abbildung 11: Untersuchung der Komprimierungsalgorithmen

Dabei hat sich gezeigt, dass JPG 95 beim Training und der Erkennung sowohl qualitativ (mean Average Precision (map)) als auch bei der Dateigröße die besten Resultate liefert, wie in Abbildung 11 dargestellt ist. Die Ergebnisse des Trainings für Autos mit Bild- und LIDAR-Daten können in Abbildung 12 gesehen werden.

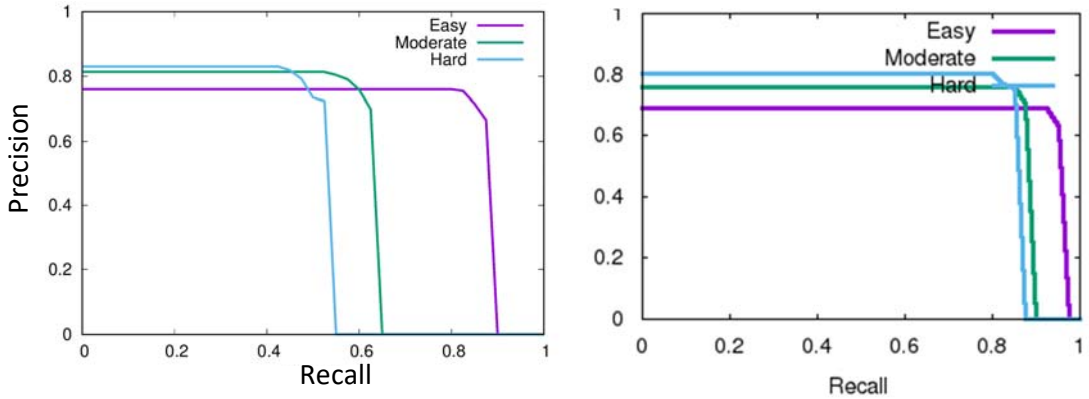

Abbildung 12: Ergebnis des Trainings für Autos mit Bild-Daten (links) und LIDAR-Daten (rechts)

3.1.3 *AP3 Systemintegration und Test*

Für die Integration wurden in **AP3.2.4** Arbeiten mit anderen Projektpartnern unternommen. Dazu werden die Ein- und Ausgangsdaten in das von Fraunhofer FOKUS entwickelte Protokoll decodiert bzw. encodiert. Die neuronalen Netze wurden den Projektpartnern IBEO, Videantis und Infineon zur Verfügung gestellt und regelmäßig aktualisiert. Gemeinsam mit den Partnern haben wir die Software auf dem FlexAISIC integriert.

Die SW-Framework-Integration für die Kartendaten wurde in **AP3.2.5** umgesetzt und mit dem Projektpartner Ibeo abgestimmt. Dazu wurden die durch die semantische Erkennung (AP 2.5.5) detektierten stangenfähigen Objekte über die entsprechenden Objektlisten-Nachrichten im von Fraunhofer FOKUS implementierten Protokoll

versendet. Die Anbindung der Schnittstelle erfolgte bei den Integrationstests im AP 3.3.

In **AP3.3.1** wurde die Integration des Gesamtsystems umgesetzt. Dazu wurden alle notwendigen Komponenten der Architektur umgesetzt, auf der Ziel-Hardware installiert und für die Erweiterungen vorbereitet. Alle Schnittstellen sind vorhanden. Die Automotive Unit des Partners Fraunhofer FOKUS wurde mit dem implementierten ZeroMQ-Protokoll ausgerüstet, das die Grundlage für alle Kommunikations-Vorgänge legt. Hierfür wird das zusammen mit dem Projektpartner Fraunhofer FOKUS eingesetzte Protokoll und die dazugehörige Bibliothek verwendet. Ebenso wurde die ROS-Verbindung zwischen Localization Unit und Automotive Unit hergestellt. Die Sensor-Anbindung an die Automotive Unit ist ebenfalls erfolgt. Dafür wurde der im Projekt entwickelte ROS2-Proxy verwendet, der die Daten der Sensoren in ROS2-Nachrichten umwandelt. Somit ist es möglich, auch ROS2-Bags mit Sensorrohdaten aufzuzeichnen und für Continuous Integration abzuspielen. Die volle Integration der Architektur kann in Abbildung 13 gesehen werden.

Das DCAITI hat im **AP3.3.2** den ROS-Proxy eingesetzt, der es erlaubt, Rohdaten im ROS2-Bag-Format aufzuzeichnen und wieder abzuspielen. Hierfür wurde eine Datenübergabe von aufgezeichneten Testdaten mit dem Projektpartner Infineon vereinbart. Die Algorithmen, die in den vorherigen Arbeitspaketen entwickelt und, bei den KI-Algorithmen, trainiert wurden, konnten somit ausgiebig getestet und validiert werden. Damit wurden die notwendigen Voraussetzungen geschaffen, um das FPGA-basierte Rapid-Prototyping-System zu integrieren.

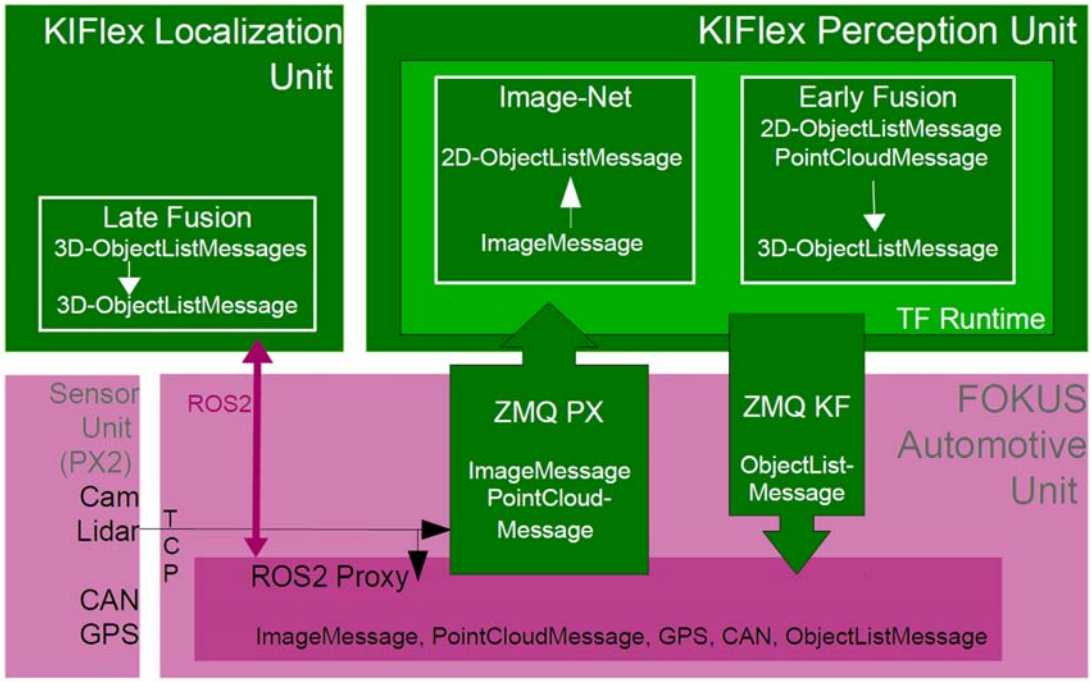

Abbildung 13: KI-FLEX Gesamtarchitektur

Die Integration des Gesamtsystems wurde in AP3.3.9 umgesetzt. Die vom Projektpartner IBEO am Versuchsfahrzeug montierten Solid-State-LIDAR-Sensoren wurden angeschlossen. Dazu wurde die Localization Unit im Fahrzeug verbaut und die zugehörige Software in Betrieb genommen. Die Anbindung an die Automotive Unit über Ethernet wurde ebenfalls realisiert. Von den Sensoren können Daten an die Automotive Unit gesendet werden. In Abbildung 14 kann der Aufbau im Fahrzeug gesehen werden.

Die Sensor Unit, an der die verschiedenen Sensoren angeschlossen sind, befindet sich im oberen Teil. Darunter befindet sich die Automotive Unit mit Hardware zur Vorverarbeitung der LIDAR-Punkte und der Bild. Die Localization Unit, an der die neuartigen Sensoren von IBEO angeschlossen sind, befindet sich in der rechten Hälfte des Bildes. Unterhalb der Automotive Unit kann der Router gesehen werden, über den alle Komponenten miteinander verbunden sind. In Abbildung 15 ist die Localization Unit nochmals detailliert zu sehen, während Abbildung 16 das Versuchsfahrzeug während der Integrationsarbeiten in der Garage des DCAITI zeigt.

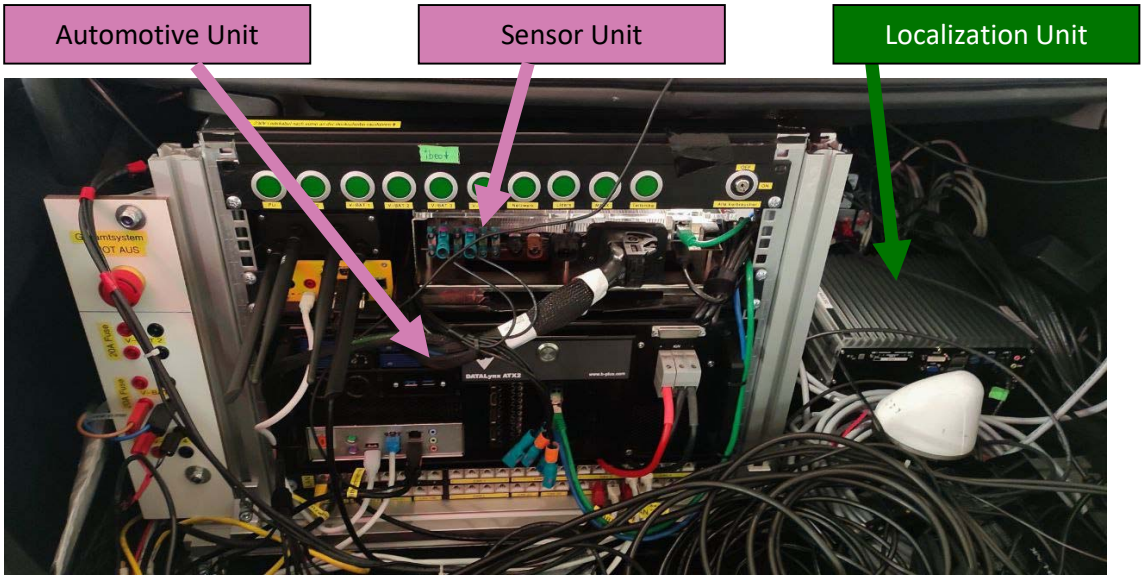

Abbildung 14: Aufbau im Fahrzeug

Abbildung 15: Localization Unit

Im Arbeitspaket 3.3.11 erfolgte die Gesamt-System-Evaluation. Dafür wurden mehrere Versuchsfahrten mit dem Versuchsträger des DCAITI durchgeführt. Bei den Versuchen wurde die Objekterkennung erprobt. Dafür wurde eine geeignete Visualisierung in 3D eingesetzt. Die Objekterkennung, in der mehrere Personen (blau), Fahrzeuge (grün) sowie Wände zu erkennen sind, ist in Abbildung 17 zu sehen.

Abbildung 16: Versuchsfahrzeug

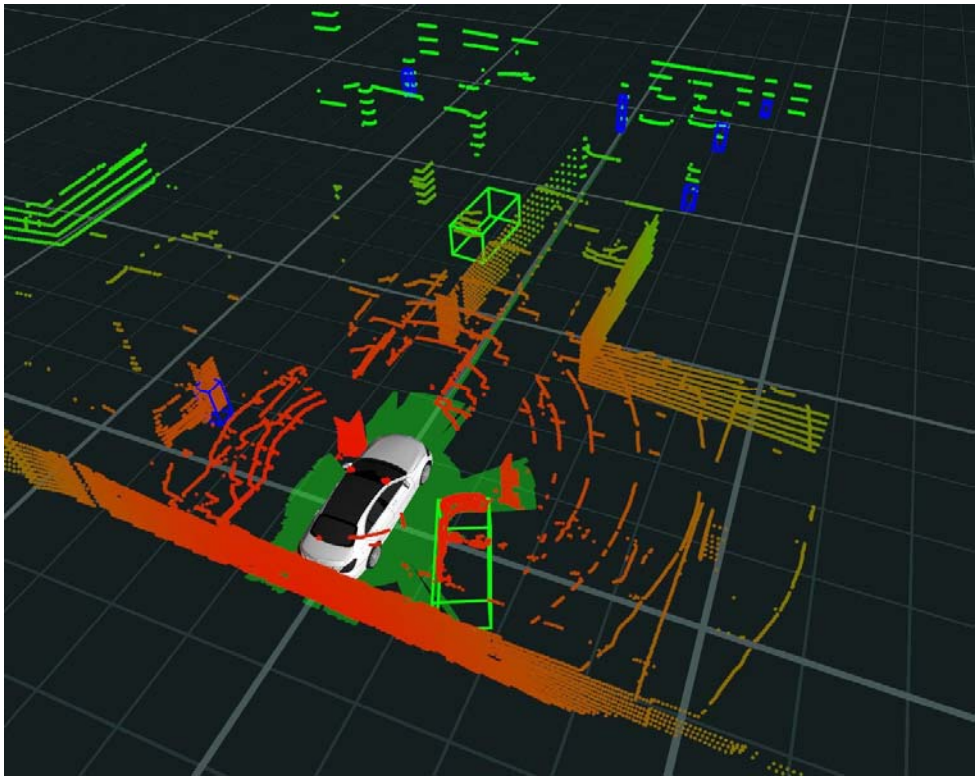

Abbildung 17: Visualisierung der Objekterkennung

3.2 Wichtigste Positionen des zahlenmäßigen Nachweises

Alle Beträge in Euro (€)

1	2	3	4
Position Gesamtfinan-zierungsplan	**Entstandene Ausgaben** insgesamt bis einschl. 2023	**Anerkannte Ausgaben** insgesamt bis einschl. 2023	**Gesamtfinanzierungsplan**
0812	324.141,31		321.993,00
0817	0,00		0,00
0820	0,00		0,00
0822	0,00		0,00
0831	0,00		0,00
0834	0,00		0,00
0835	1.434,78		2.000,00
0843	1.370,38		5.388,00
0846	281,40		5.800,00
0850	0,00		9.950,00
Summe:	327.227,87		345.131,00

			Nachgewiesen	**Anerkannt**
14	Entstandene Ausgaben insgesamt **(Summe von Spalte 2)**		327.227,87	
15	Anteil Eigenmittel lt. Finanzierungsplan		0,00	
16	Mittel Dritter und Einnahmen lt. Finanzierungsplan ohne Zeile 17		0,00	
17	**Weitere** Mittel Dritter und Einnahmen ohne Zeile 16 (gem. NB)	17a) Gesamt 0,00	17b) Bundesanteil [1] 0,00	
18	Verbleibender Anteil des Bundes (14 ./. 15 ./. 16 ./. 17b)		327.227,87	
18a	Projektpauschale		65.445,57	
19	Zahlung auf Anteil des Bundes		393.363,24	
20	Kassenbestand am 31.08.2023 (19 ./. 18 / 18a)		689,80	

3.3 Notwendigkeit und Angemessenheit der geleisteten Arbeit

Die im Projekt KI-FLEX geleistete Arbeit sowie die dafür aufgewendeten Budgemtittel waren notwendig und angemessen. Die im Projektantrag gestellten Ziele wurden mit den aufgewendeten Mitteln erreicht. Darüber hinaus konnten kostenneutral die Aktivitäten verlängert werden, um die Projektergebnisse auch auf der Projekthardware zu testen und zu validieren. Die zusätzliche Zeit wurde dafür genutzt, die Szenarien vorab auf alternativer Hardware, aber mit der Konfiguration für die Zielhardware, zu testen.

3.4 Verwertbarkeit der Ergebnisse

Die in KI-FLEX erzielten Ergebnisse werden in der Automobilindustrie in zukünftigen Hardware- und Software-Entwicklungen für automatisierte Fahrfunktionen eingesetzt werden können.

- Erkennung von dynamischen Objekten mit Kamera
- Erkennung von statischen Objekten mit Kamera
- Erkennung von dynamischen Objekten mit Kamera
- Erkennung von statischen Objekten mit Kamera
- Flexible, herstellerunabhängige Kartendefinition und -schnittstellen.

- Semantische Segmentierung von Bildern und Punktwolken
- Semantischer SLAM, basierend auf der Semantischen Segmentierung
- Verfahren, um Training und Inferenz von Bilddaten auf komprimierten Bildern auszuführen
- Modularität und Kommunikationsframework für zukünftige Architekturen im Bereich automatisiertes Fahren
- Ausführung von neuronalen Netzen auf einem FlexASIC
- Ausführung von neuronalen Netzen auf einem rekonfigurierbaren FPGA
- Visualisierung sowie Auswertungs- und Validierungstools für weitere Forschungsprototypen und Serienproduktion
- Detektion mit neuartigen Solid State-LIDAR-Sensoren

3.5 Fortschritte auf dem Gebiet des Vorhabens bei anderen Stellen

Nach Beginn des Projektes KI-FLEX wurde mit dem Projekt ANDANTE ebenfalls ein Projekt zur Entwicklung von dedizierter Hardware für neuronale Netze gestartet. In diesem Projekt wurden anwendungsspezifische Elektroniksysteme entwickelt. Über die reinen Prozessorbausteine hinaus beschäftigte sich dieses Projekt auch mit Speicher oder zusätzlichen Rechenblöcken. An ANDANTE waren ebenfalls namhafte Industriepartner wie Valeo, Heimann Sensor oder eesy-innovation sowie renommierte Wissenschafts-einrichtungen wie die Fraunhofer-Institute IPMS, EMFT und IIS, die FAU Erlangen oder die Technische Universität Dresden beteiligt.

3.6 Geplante Veröffentlichungen

- Beitrag in „Internationales Verkehrswesen“

Kurzer inhaltlicher Bericht

1. Verwendung der Zuwendung und erzielte Ergebnisse

1.1. System zur Umfeldwahrnehmung

Für die Umfeldwahrnehmung wurde zunächst ein Framework aufgesetzt zum Trainieren der Neuronalen Netze. Für die Erkennung von Objekten wurde TensorFlow eingesetzt, da die in KI-FLEX entwickelte Hardware die TensorFlow API unterstützt. Es wurde sowohl ein Netzwerk zur Erkennung in Bildern als auch in Lidar-Daten entwickelt. Für die Bilderkennung wurde ein Single-Shot-Detector eingesetzt und Inception V2 als Feature Extractor genutzt. In der Verarbeitung werden die Bilder auf eine einheitliche Größe verkleinert, die auch von der KI-FLEX Hardware verarbeitet werden kann. Anschließend werden ein Base Network und ein Feature Network eingesetzt, um Features zu detektieren. Nun wird das Prediction Network ausgeführt, welches hauptsächlich aus Convolutional Layern besteht. Dabei wird eine Klassifizierung mit Konfidenzen durchgeführt und die Objekte werden durch Bounding Boxes im Bild markiert. Im Post Processing werden mit der Non maximum Suppression Methode überlappende Bounding Boxes entfernt, bevor die Daten wieder auf die ursprüngliche Pixelgröße projiziert. Schließlich werde die Pixelkoordinaten in Koordinaten in der realen Welt umgerechnet. Bei der LIDAR-Erkennung werden die Daten so aufbereitet, dass sie mit neuronalen Netzen verarbeitet werden können. Zu diesem Zweck werden die vom Lidar-Sensor detektieren Punkte in Zellen mit einer Größe von 15 cm gesammelt und zu jeder Zelle ausgewertet, auf welcher Höhe sich der höchste und der niedrigste Punkt pro Zelle befinden. Zusätzlich wird die maximale Reflektion in einer Zelle bestimmt. Somit entsteht ein Occupancy Grid von 1000x1000 Zellen. Mit den drei Werten wird ein Drei-Kanal-Bild erstellt werden, bei dem die Werte auf die RGB-Kanäle des Bildes gelegt werden. Für das Training der Netze wurde ein ähnlicher Ansatz wie für die Bild-Daten gewählt. Zunächst wird in einem eigenen Modul im Pre-Processing die Occupancy-Grid-Darstellung berechnet. Durch die drei Netzwerk-Schritte Base Network, Feature Network und Prediction Network werden Features berechnet und abschließend Objektklassen und Bounding Boxes ausgegeben.

1.2. Semantischer SLAM

Für den semantischen SLAM wurden die Algorithmen zur semantischen Markierung (Labelling) von Bilddaten entwickelt. Dazu wurde zunächst eine Netzwerkarchitektur entwickelt, die auf dem Mask-Scoring RCNN Algorithmus basiert. Die eigentliche Lokalisierung basiert auf einem Partikelfilter, der als Input die stangenartigen Objekte aus der Semantischen Erkennung sowie die Karten aus der Kartenschnittstelle verwendet. Dazu werden Partikel auf der Karte verteilt und anhand der Abstände zu den Objekten aus der Semantischen Erkennung die Probabilität für die einzelnen Partikel berechnet. Ab einem Schrankenwert der Wahrscheinlichkeit wird ein Partikel als Aufenthaltsort des Fahrzeugs akzeptiert.

1.3. Systemintegration und Test

Für die Gesamtsystemintegration wurden Arbeiten mit anderen Projektpartnern durchgeführt. Die neuronalen Netze wurden regelmäßig aktualisiert und den Partnern IBEO, Videantis und Infineon zur Verfügung gestellt. Gemeinsam mit den Partnern wurde die Integration der Software auf dem FlexAISIC durchgeführt.

Die SW-Framework-Integration für die Kartendaten wurde mit dem Partner Ibeo abgestimmt. Die durch die semantische Erkennung detektierten Objekte wurden über Objektlisten-Nachrichten versendet.

Anschließend wurde die Integration des Gesamtsystems umgesetzt. Die Schnittstellen wurden implementiert basierend auf dem ZeroMQ-Protokoll. Die ROS-Verbindung zwischen der Localization Unit und der Automotive Unit wurde hergestellt. Die Sensor-Anbindung an die Automotive Unit wurde ebenfalls realisiert. Die Neuronalen Netzen wurden ausgiebig getestet und validiert. Dadurch wurden die Voraussetzungen für die Integration des FPGA-basierten Rapid-Prototyping-Systems geschaffen.

Im Rahmen der Integration wurden die Solid-State-LIDAR-Sensoren von IBEO am Versuchsfahrzeug angeschlossen. Die Localization Unit wurde im Fahrzeug verbaut und in Betrieb genommen sowie an die Automotive Unit angebunden. Abschließend fand die Gesamt-System-Evaluation statt. Versuchsfahrten mit dem Versuchsträger des DCAITI wurden durchgeführt, um die Objekterkennung zu testen. Eine Visualisierung in 3D wurde verwendet, um die Detektion von Personen, Fahrzeugen und Wänden darzustellen.