Digitale Bildverarbeitung auf einem selbst konstruierten Modellfahrzeug

Masterarbeit

im Studiengang

Informatik

vorgelegt von

Georg Jenschmischek

am 09. Dezember 2016

an der Hochschule für Technik, Wirtschaft und Kultur Leipzig

Erstprüferin:	Prof. Dr. Sibylle Schwarz
Zweitprüfer:	Prof. Dr. Klaus Bastian

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quelltextverzeichnis

Abkürzungsverzeichnis

AADC	…	Audi Autonomous Driving Cup
ADTF	…	Automotive Data and Time Triggered Framework
BASt	…	Bundesanstalt für Straßenwesen
CISC	…	Complex Instruction Set Computer
HTWK	…	Hochschule für Technik, Wirtschaft und Kultur
IEEE	…	Institute of Electrical and Electronics Engineers
IMN	…	Informatik, Mathematik und Naturwissenschaften
IMU	…	Inertiale Messeinheit, engl. inertial measurement unit
IPM	…	Inverse Perspective Mapping
KI	…	Künstliche Intelligenz
LiDAR		Laser-Radar, engl. Light detection and ranging
NHTSA	…	National Highway Traffic Safety Administration
OpenCL	…	Open Computing Language
OpenCV	…	‚Open Computer Vision'-Biblitohek
RISC	…	Reduced Instruction Set Computer
ROS	…	Robot Operating System
USB	…	Universal Serial Bus

Abstract

Diese Arbeit beschäftigt sich zunächst mit der Konstruktion eines Modellautos, das in der Lage sein soll, sich autonom über eine Miniatur-Straßenlandschaft zu bewegen. Nach einer Anforderungsanalyse wird die verwendete Hardware vorgestellt und begründet, warum sich diese für das Modellfahrzeug eignet. Es wird außerdem eine Liste an Verbesserungsvorschlägen präsentiert, die nach Konstruktion und Test des Autos entstanden ist.

Im Anschluss wird für das konstruierte Modellfahrzeug ein Spurerkennungsalgorithmus entwickelt. Dazu wird ein spatiotemporaler Ansatz implementiert, der sich nach einer Anforderungsanalyse als geeignet erwiesen hat. Diese Implementation wird an die Gegebenheiten der Miniatur-Straßenlandschaft angepasst. Abschließend werden Anforderungserfüllung und Grenzen des entwickelten Algorithmus anhand eines Praxistests dargestellt.

Schlagwörter: Autonomes Fahren, digitale Bildverarbeitung, Spurerkennung, Spurverfolgung, Modellfahrzeug, Fish-Eye-Kamera, Weitwinkelkamera, Spatiotemporal

1 Einleitung

Im Film ‚I, Robot' aus dem Jahr 2004 ([VG04]) wird dargestellt, wie Menschen in der nahen Zukunft ihre Autos nicht mehr selbst fahren müssen. Stattdessen bewegen sich die Fahrzeuge eigenständig zum angegebenen Ziel, die Insassen können sich mit anderen Tätigkeiten beschäftigen.

Was vor zwölf Jahren noch als Utopie in einem Science-Fiction-Film präsentiert wurde, ist heutzutage schon fast Realität. Große Fahrzeughersteller wie Audi, BMW oder Tesla forschen intensiv an Fahrassistenzsystemen, die im Zusammenspiel autonome Fahrfunktionen ermöglichen können.

Auch an der Hochschule für Technik, Wirtschaft und Kultur Leipzig existiert eine studentische Forschungsgruppe für autonomes Fahren. Die vorliegende Arbeit entstand im Rahmen dieser Forschungsgruppe und beschäftigt sich mit dem Aufbau eines autonomen Modellfahrzeugs sowie der Entwicklung eines Spurerkennungsalgorithmus. Die genauen Ziele dieser Arbeit werden im Folgenden erläutert.

1.1 Zielstellung

Die Zielstellung dieser Arbeit besteht aus zwei Hauptaufgaben. Zuerst soll für die studentische Forschungsgruppe für autonomes Fahren der Hochschule für Technik, Wirtschaft und Kultur Leipzig ein autonomes Modellfahrzeug entworfen und konstruiert werden. Dieses soll mit Sensorik, Aktorik und Rechentechnik ausgestattet werden, sodass es sich selbstständig über eine unbekannte Miniatur-Straßenlandschaft bewegen kann. Dazu muss zunächst geklärt werden, welche Anforderungen an ein solches Fahrzeug gestellt werden. Bei der Umsetzung dieser Anforderungen wird anschließend abgewogen, welche Hardware diese erfüllen kann.

Im zweiten Teil dieser Arbeit soll ein Spurerkennungsalgorithmus für das zuvor konstruierte Modellauto entwickelt werden. Auch hier müssen zunächst Anforderungen festgelegt werden, die die Auswahl an Spurerkennungsansätzen eingrenzen. Anschließend soll ein ausgewählter Spurerkennungsansatz umgesetzt und für das neue Modellfahrzeug angepasst werden.

2 Grundlagen

Zu Beginn dieser Arbeit sollen zunächst theoretischen Grundlagen erläutert werden. Deshalb wird im folgenden Kapitel der Begriff des autonomen Autofahrens definiert. Neben der geschichtlichen Entstehung der autonomen Fahrzeuge soll außerdem auf aktuelle Entwicklungen eingegangen und dabei die Forschungsarbeit an der Hochschule für Technik, Wirtschaft und Kultur (HTWK) näher beleuchtet werden.

Der zweite Teil dieses Kapitels beschäftigt sich mit der digitalen Bildverarbeitung. Auch hier sollen zu Beginn Begrifflichkeiten und Methoden geklärt werden. Anschließend sollen Einsatzmöglichkeiten von Bildverarbeitungsverfahren beim autonomen Autofahren erläutert und dadurch der Zusammenhang zwischen den beiden Themengebieten dieser Arbeit aufgezeigt werden.

2.1 Autonomes Fahren

2.1.1 Begriffsdefinition

Die Definition des Begriffs ‚Autonomes Fahren' stellt sich als schwierig heraus, da keine international anerkannte, einheitliche Bestimmung dieses Begriffs existiert. Dementsprechend sollen im Folgenden zwei Definitionsansätze präsentiert werden, die gemeinsam ein Gesamtbild des ‚Autonomen Fahrens' erzeugen.

Ein erster Definitionsansatz kann dadurch erfolgen, dass die beiden Begriffe ‚Autonom' und ‚Fahren' getrennt betrachtet und deren Bedeutung erläutert wird. Das Wort ‚Autonom' leitet sich aus den zwei griechischen Wörtern ‚αὐτός' (autos) und ‚νόμος' (nomos) ab, welche übersetzt ‚selbst' und ‚Gesetz' bedeuten. Im Deutschen entspricht ‚Autonom' den Worten ‚unabhängig' bzw. ‚eigenständig'. (vgl. [Du14])

In dieser Arbeit wird der Begriff ‚Fahren' gleichgesetzt mit dem Führen von Personen- bzw. Lastkraftwagen im alltäglichen Straßenverkehr. Dazu zählen das Fahren durch Stadtgebiete, auf Landstraßen und Autobahnen bei Tag bzw. Nacht. Besondere Fahrzeuge, wie bspw. Gabelstapler, oder besondere Fahrregionen, wie bspw. Waldwege, sollen einfachheitshalber ausgeschlossen werden.

Kombiniert man nun also die Bedeutungen der Wörter ‚Autonom' und ‚Fahren' im eben erläuterten Sinne, so ergibt sich auch die Intension der zusammengesetzten Wortgruppe ‚Autonomes Fahren', welche für diese Arbeit gelten soll: Die eigenständige Führung eines Personen- bzw. Lastkraftwagen. Genauer gesagt bedeutet das, dass sich ein Fahrzeug ohne Aktionen eines Fahrers selbständig im Straßenverkehr bewegen kann.

Anders versuchen [JM15] den Begriff ‚Autonomes Fahren' zu erläutern. Sie sehen das autonome Fahren als eine logische Fortsetzung der bisherigen Fahrzeugentwicklung. In den letzten Jahren wurden immer mehr Systeme in Autos eingebaut, die den Fahrer bei Führung des Fahrzeugs unterstützen sollen. Dazu zählen z.B. Spurstabilitätssysteme, Tempomaten, das Antiblockiersystem oder Einparkhilfen. Kombiniert man diese Systeme oder erweitert sie bspw. mit Kameras, so liegt laut den Autoren die Annahme nah, dass ein Auto bald keinen Fahrer mehr benötigen könnte.

Beide Definitionen für ‚Autonomes Fahren' sind für diese Arbeit relevant. Die erste Definition beschreibt allgemein die Tätigkeit des ‚Autonomes Fahrens', während die zweite Definition die technische Weiterentwicklung von Hard- und Software betrachtet. Das autonome Modellfahrzeug, dass im Zuge dieser Arbeit entstehen soll, wird diesen Definitionen folgend derart mit Hard- und Software ausgestattet, dass es die Tätigkeit des ‚Autonomen Fahrens' erfüllen kann.

Im folgenden Kapitel wird auf die aktuellen Fortschritte im Bereich des ‚Autonomen Fahrens' und die Arbeit der HTWK auf diesem Gebiet eingegangen.

2.1.2 Aktuelle Entwicklungen

Im vorangegangenen Abschnitt wurde der Begriff des autonomen Fahrens für diese Arbeit definiert. Nun soll erläutert werden, welche aktuellen Entwicklungen sich in diesem Forschungsbereich ereignen. Außerdem werden Informationen über die Forschungsgruppe ‚HTWK Smart-Driving' dargestellt.

Übereinstimmend geben [We14] und [Vi15] an, dass der erste Forschungsdurchbruch im Bereich des selbstfahrenden Automobils die Arbeit von Ernst Dickmann in den Jahren 1986 bis 1994 gewesen ist. In dieser Zeit stattet er an der Bundeswehr-Universität München einen Mercedes Van mit mehreren Kamerasystemen aus. Mit diesem fuhr er hunderte Kilometer autonom auf der Autobahn bei durchschnittlich 110 Kilometer pro Stunde. Dabei füllte die Rechentechnik, welche die Kamerabilder auswertete, noch den gesamten Rückraum des Kleintransporters aus.

Trotz dieses Durchbruchs in der Forschung auf diesem Gebiet wagten es die Automobilfirmen zunächst nicht, die Arbeit von Ernst Dickmann weiterzuentwickeln. Laut [Vi15] wollten sie die Autofahrer nicht bevormunden. Deshalb wurde lange Zeit keine Forschungsarbeit für das autonome Fahren verrichtet.

Erst in den vergangenen zehn Jahren kam wieder Bewegung in die Entwicklung rund um das autonome Fahrzeug. Dabei tat sich insbesondere Google ab dem Jahr 2008 als treibende Kraft hervor (vgl. [We14]). Inzwischen haben auch die Automobilhersteller ihre Meinung zur Thematik des autonomen Fahrens geändert und so forscht bspw. auch Audi in diesem Bereich.

Auch an der HTWK Leipzig beschäftigt sich eine Forschungsgruppe mit der Thematik des autonomen Fahrens. Das sogenannte ‚Team HTWK Smart-Driving' nimmt seit 2014 am ‚Audi Autonomous Driving Cup' (AADC) teil. In diesem Wettbewerb, welchen der namensgebende Automobilhersteller veranstaltet, wird Studentengruppen ein Modellauto im Maßstab 1:8 zur Verfügung gestellt. Auf diesem befinden sich Sensoren, wie Kameras oder Ultraschallsensoren, sowie ein Mini-Computer, der die Sensordaten verarbeiten kann. Wie die anderen Wettbewerbsteilnehmer arbeitet das ‚Team HTWK Smart-Driving' daran, Software für das Modellfahrzeug zu entwickeln, damit dieses sich selbstständig über eine im Vorfeld unbekannte Miniatur-Straßenlandschaft bewegen kann. Nach einem auf ca. sechs Monate begrenzten Zeitraum treffen sich alle teilnehmenden Studentengruppen in Ingolstadt und testen, welches Modellauto am sichersten autonom fährt.

Kamerasysteme sind ein wichtiger Sensor in allen autonomen Fahrzeugen. Die Auswertung und Interpretation der Bildinformationen kann mithilfe digitaler Bildverarbeitung erfolgen. Was sich hinter diesem Begriff verbirgt und wofür digitale Bildverarbeitung bei selbstfahrenden Autos eingesetzt werden kann, wird im folgenden Abschnitt erläutert.

2.2 Digitale Bildverarbeitung

2.2.1 Begriffsdefinition

„Ein Bild sagt mehr als tausend Worte.“, ist ein bekanntes Sprichwort in der deutschen Sprache. Tatsächlich lässt sich mit diesem Satz der Inhalt der Methodik der digitalen Bildverarbeitung kurz und knapp beschreiben. Ziel der digitalen Bildverarbeitung ist es, aus Bilddaten jeglicher Art Informationen einer höheren Abstraktionsebene zu gewinnen. Damit hat sie sich zum Standardwerkzeug für praktisch alle Naturwissenschaften und technischen Disziplinen entwickelt (vgl. [Jä05]).

Die Verfahren, die in der digitalen Bildverarbeitung angewendet werden, haben alle das Ziel, die vom Anwender geforderten Informationen besser erkennbar zu machen. Deshalb ist es nicht ungewöhnlich, dass verschiedene Anwender auf das gleiche Bild verschiedene Verfahrensketten anwenden, um genau die Merkmale hervorzuheben, die sie interessieren. Die eigentliche Informationsgewinnung erfolgt, ausgehend vom verarbeiteten Bild, meist mit Methoden aus dem Bereich der Mustererkennung, welcher eng verwandt mit der digitalen Bildverarbeitung ist. (vgl. [GW08])

Im folgenden Kapitel werden kurz die Verfahren der digitalen Bildverarbeitung erläutert, die im Rahmen dieser Arbeit zum Einsatz gekommen sind.

2.2.2 Verwendete Verfahren

In diesem Abschnitt sollen die in dieser Arbeit verwendeten Verfahren der digitalen Bildverarbeitung erklärt werden. Dazu zählen der Hough-Algorithmus, das Inverse Perspective Mapping sowie die Binarisierung von Bildern.

Zunächst soll aber der Begriff ‚Bild‘ formal definiert werden. Ein Bild B kann als Funktion betrachtet werden, die aus der Menge der Bildpositionen pos in die Menge der Farben col abbildet.

$$B: pos \rightarrow col$$

Habe das Bild die Höhe h_B und die Breite b_B so sei die Positionenmenge:

$$pos = \{0, \dots, b_B - 1\} \times \{0, \dots, h_B - 1\}$$

Ein Element der Positionenmenge wird in dieser Arbeit anhand seiner x- und y-Koordinaten $(x, y) \in pos$ angegeben.

Die Menge der Farben kann je nach Farbraum des Bildes unterschiedlich definiert sein. Ein graues Bild, welches eine Farbtiefe von n_{col}-Bit besitzt, habe eine Farbmenge:

$$col_{grau} = \{0, \dots, 2^{n_{col}} - 1\}$$

Dabei entspricht der Wert 0 der Farbe Schwarz, der Wert $2^{n_{col}} - 1$ der Farbe Weiß.

Farbbilder können in der digitalen Bildverarbeitung dargestellt werden, in dem statt einem Farbraum drei Farbräume verwendet werden. Dazu werden drei graue Farbräume verknüpft, wobei jeder der drei grauen Farbräume im Farbbild einem roten, grünen bzw. blauen Farbraum entspricht. Durch Kombination verschiedenstarker Rot-, Grün- bzw. Blautöne kann jede beliebige Farbe dargestellt werden. Es ergibt sich:

$$col_{RGB} = \{0, \dots, 2^{n_{col}} - 1\}^3$$

Da die in dieser Arbeit vorgestellten Algorithmen nicht mit Farb- sondern nur mit Helligkeitsinformationen arbeiten, wird ab sofort standardmäßig von Grauwertbildern ausgegangen.

2.2.2.1 Hough-Algorithmus

Der Hough-Algorithmus ist ein Verfahren, das gerade Linien in Bildern identifizieren kann. Es zählt zu den Verfahren der Segmentation, bei denen nach Bereichen im Bild gesucht wird, die bestimmte, bekannte Eigenschaften besitzen. (vgl. [Jä05] S. 481f.) In diesem Fall wäre die bekannte Eigenschaft die Geradlinigkeit.

Der Hough-Algorithmus beruht grundsätzlich darauf, dass Geraden in der Hesse-Normalform dargestellt werden können:

$$g: d = x * \cos\phi + y * \sin\phi$$

Dabei ist die Gerade g dargestellt durch den kürzesten Abstand zum Koordinatenursprung $d \in \mathbb{R}_{\geq 0}$ sowie durch den Winkel der Geraden zur x-Achse $\phi \in [0, \pi]$. In der Positionenmenge pos wären x und y Variablen, während ϕ und d Parameter sind. Der Hough-Raum H dreht diese Beziehung um und sei definiert als:

$$H = [0, \pi] \times \mathbb{R}_{\geq 0}$$

Er besitzt also statt einer x- bzw. y-Achse eine ϕ- bzw. d-Achse. Demnach entspricht ein Punkt im Hough-Raum genau einer Geraden im pos-Raum.

Der Hough-Algorithmus generiert nun eine sogenannte Voting-Matrix V, in der jeder Punkt (also jede Gerade in pos) eine Menge von Votes erhalten kann. Sei $P \subseteq pos$ die Menge aller Bildpunkte, die die Eigenschaften der gesuchten Geraden erfüllen, also bspw. besonders hell oder dunkel sind. V sei dann definiert als

$$V: H \to \mathbb{N}$$

$$V(\phi_V, d_V) = \sum_{(x_P, y_P) \in P} \begin{cases} 1, f\ddot{u}r\ d_V = x_P * \cos\phi_V + y_P * \sin\phi_V \\ 0, sonst \end{cases}$$

Alle Punkte aus P erzeugen also einen Vote in V für alle Geraden, die durch sie hindurchgehen. Die stärksten Geraden im Bild sind dann die globalen Maxima von V und die Ergebnisse des Hough-Algorithmus.

Abbildung 1 zeigt das Ergebnis des Hough-Algorithmus. Links ist das Ausgangsbild zu sehen, rechts die Votingmatrix V. Die zwei hellen Punkte in V, die die meisten Votes erhalten haben, sind gut zu erkennen. Diese entsprechen den zwei Geraden aus dem linken Bild.

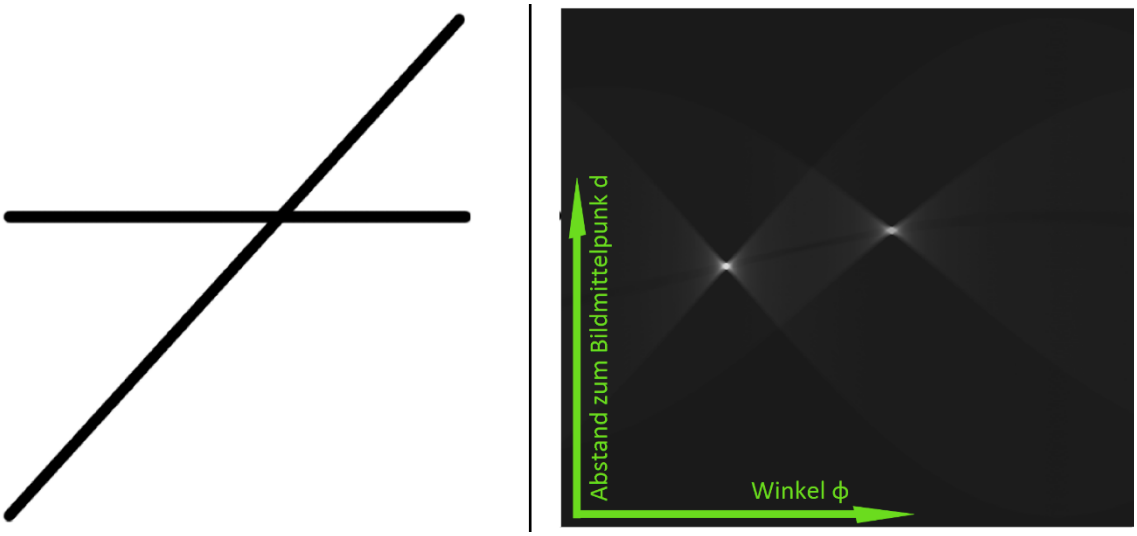

Abbildung 1 – Hough-Algorithmus-Beispiel.
Ausgangsbild links, Votingmatrix V rechts (angepasst aus [Da06])

Die Ergebnisse des Hough-Algorithmus können beeinflusst werden, indem die Menge an relevanten Bildpunkten P anders eingeschränkt wird. Außerdem können die Wertebereiche und die Auflösung von H in V reduziert werden, um bestimmte Ergebnisgeraden auszuschließen bzw. zusammenzufassen.

Der Hough-Algorithmus wird in dieser Arbeit sowohl in der bisher verwendeten (vgl. Kapitel 4.1.1.1) als auch in der neu entworfenen Spurerkennung (vgl. Kapitel 4.4.2.2) angewandt.

2.2.2.2 Inverse Perspective Mapping (IPM)

Das Inverse Perspective Mapping (IPM) ist eine perspektivische Transformation. Perspektivische Transformationen sind Funktionen, die die perspektivisch verzerrten Bildpositionen pos auf gewünschte, entzerrte Bildpositionen pos' abbilden. Inverse Perspective Mapping ist dabei die Projektion der Bildszene von der Bildebene auf eine Ebene der realen Welt, bei der das Projektionszentrum erhalten bleibt (vgl. [Fa90] S. 614). Im Falle des autonomen Fahrens kann IPM bspw. eingesetzt werden, um das perspektivische Bild einer Frontkamera so zu entzerren, dass es aussieht, als würde man von oben auf die Straßenszene schauen (Vogelperspektive).

Eine Möglichkeit zur Bestimmung einer eindeutigen IPM-Abbildungsfunktion wird in Kapitel 4.1.1.2 erläutert. Dort wird die Projektion des Fahrzeugkamerabildes auf die Straßenebene unter Zuhilfenahme des Lochkameramodells beschrieben.

2.2.2.3 Binarisierung

Unter dem Begriff Binarisierung wird in der digitalen Bildverarbeitung eine Funktion $f_{bin}: B \rightarrow B_{bin}$ verstanden, die ein Bild mit dem Farbraum $col = \{0, \dots, 2^{n_{col}} - 1\}$ in ein Bild mit dem binären Farbraum $col_{bin} = \{0,1\}$ abbildet. Ein Bild, dass einer Binarisierung unterzogen wurde, besteht im Anschluss also nur noch aus den Farben Schwarz (Farbwert: 0) und Weiß (Farbwert: 1).

Binarisierungen nutzen dabei einen Grenzwert $g \in col$. Alle Farbwerte oberhalb des Grenzwertes werden weiß gefärbt, alle darunter schwarz.

$$B_{bin}(x,y) = f_{bin}(B)(x,y) = \begin{cases} 0, f\ddot{u}r\ B(x,y) < g \\ 1, sonst \end{cases}$$

Abbildung 2 zeigt ein Grauwertbild mit einer Farbtiefe von 8 Bit mit $col_8 = \{0, \dots, 255\}$ sowie das dazugehörige binarisierte Bild bei einem Grenzwert $g = 205$.

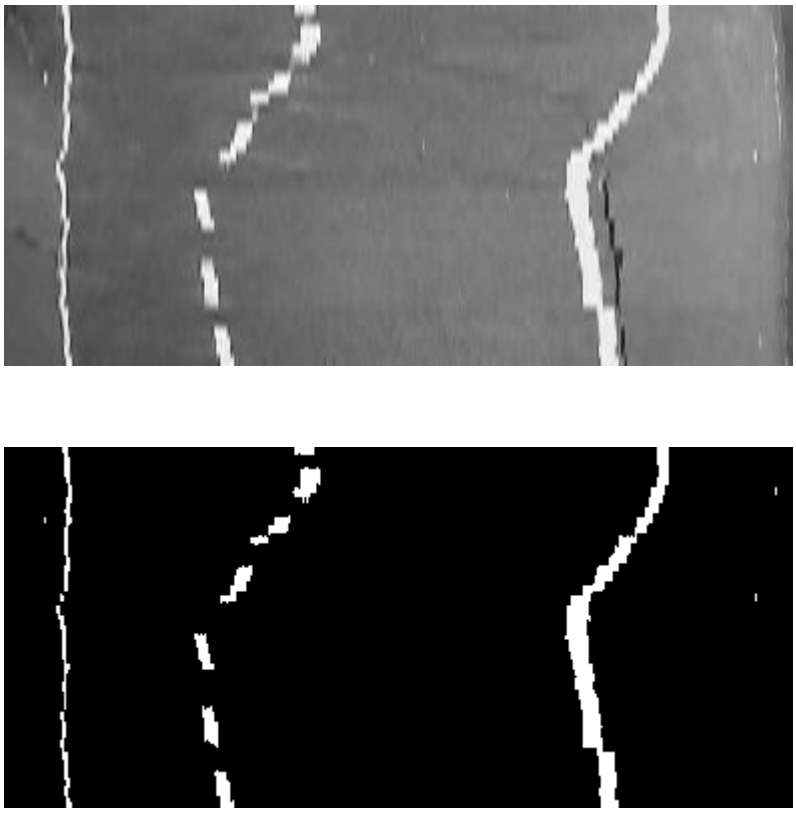

Abbildung 2 – Binarisierungsbeispiel
Oben: Grauwertbild. Unten: Binarisertes Bild bei Grenzwert $g = 205$

Der Grenzwert kann statisch festgelegt werden und damit für die gesamte Binarisierung unverändert sein. Er kann aber auch dynamisch anhand verschiedener Parameter, wie der Umgebung des aktuellen Bildpunktes, Helligkeitsverhältnisse oder dem zeitlichen Verlauf von diesen Parametern, berechnet werden. Dynamische Grenzwerte können im Gegensatz zu statischen Grenzwerten auf Veränderungen innerhalb eines Bildes oder zwischen verschiedenen Bildern reagieren. Dafür benötigen statische Grenzwerte keine zusätzliche Berechnungszeit.

Eine Methode zur dynamischen Binarisierung, die in dieser Arbeit verwendet wird, ist der Yen-Algorithmus aus [YCC95]. Dieser wird im bisher verwendeten Spurerkennungsalgorithmus (vgl. Kapitel 4.1.1.3) sowie im neu entwickelten Spurerkennungsalgorithmus (vgl. Kapitel 4.4.2.2) verwendet.

Binarisierungen werden, wie der Hough-Algorithmus, zu den Segmentationen gezählt. Sie fassen Bereiche zusammen, die bestimmte Helligkeitsaspekte aufweisen. Allgemein werden sie zur Kontrastverstärkung eingesetzt, indem eine Farbe die interessanten Bildbereiche beschreibt, während die andere den Hintergrund abbildet (vgl. [CSS14] S. 1f.). Beim autonomen Fahren können so bspw. helle Spurmarkierungen (Farbwert: 1) vor dem Straßenuntergrund (Farbwert: 0) hervorgehoben werden.

Diesem Gedanken folgend wird im Anschluss an dieses Kapitel der Zusammenhang zwischen der digitalen Bildverarbeitung und dem autonomen Fahren hergestellt.

2.3 Digitale Bildverarbeitung beim autonomen Fahren

Wie in Kapitel 2.2.1 erläutert, besteht das Ziel der digitalen Bildverarbeitung darin, abstrakte Informationen aus Bilddaten zu gewinnen. Um dieses Ziel zu erreichen, werden Verfahren, wie jene aus dem vorangegangenen Abschnitt, auf das Datenmaterial angewendet und dadurch die gewünschten Bildmerkmale extrahiert. Wofür die digitale Bildverarbeitung im Bereich des autonomen Fahrens eingesetzt werden kann, soll im Folgenden kurz erläutert werden.

In autonomen Fahrzeugen sind verschiedene Sensoren verbaut, welche die menschlichen Sinne eines Fahrers ersetzen sollen. In vielen selbstfahrenden Autos gehören dazu auch optische Sensorsysteme. Diese können einerseits zur reinen Abstandsbestimmung genutzt werden, wie Infrarotsensoren oder die darauf basierenden Laser-Radar-Systeme (LiDAR, engl. Light detection and ranging). Andererseits zählen zu den optischen Sensoren auch die Kamerasysteme. Diese unterscheiden sich je nach autonomem Fahrzeug in ihrer Anzahl, Position oder ihren Linsensystemen. Allen gemein ist allerdings, dass sie eine Bildfolge der Fahrzeugumgebung generieren.

Wird in dieser Bildfolge nun nach bestimmten Informationen gesucht, so kann dies mit Methoden der digitalen Bildverarbeitung erfolgen. Eine Aufgabe, die nur mithilfe von Kameras gelöst werden kann, ist die Erkennung und Verfolgung von Fahrspurmarkierungen. Da diese Markierungen zumeist flach auf die Straße geklebt oder sogar nur gemalt sind, können sie nur auf Bildern detektiert werden. Aus diesem Grund werden Algorithmen zur Spurerkennung und -verfolgung mit Verfahren der digitalen Bildverarbeitung realisiert.

Weitere Anwendungsfälle der digitalen Bildverarbeitung auf Kamerabildern ist die Erkennung von Straßenschildern und Ampeln. Auch diese beiden Verkehrshinweise sind zurzeit nur über das Kamerabild erfassbar. In absehbarer Zukunft könnten Ampeln und Verkehrsschilder aber auch mit den Fahrzeugen kommunizieren und ihre Erkennung so überflüssig machen (vgl. [Ha15]).

Zusätzlich zu den Informationen, welche das autonome Fahrzeug nur über die Kamera wahrnehmen kann, können mit Methoden der digitalen Bildverarbeitung auch Erkenntnisse über Hindernisse, Straßenverhältnisse oder Orientierung des Fahrzeugs gewonnen werden. Andere Sensorsysteme am selbstfahrenden Auto können diese Informationen ebenfalls erzeugen. Vereinigt man die Informationen aus verschiedenen Sensoren miteinander, so können Fehlmessungen einzelner Sensoren ausgeglichen werden. Diese Technik wird Sensorfusion genannt.

Um Methoden der digitalen Bildverarbeitung im autonomen Fahren zu verwenden, müssen entsprechende Verfahren getestet werden können. Dies kann bspw. mithilfe präparierter Videos geschehen, die die Videokameras als Sensor simulieren. Mit fortschreitender Entwicklung ist es jedoch ratsam Algorithmen auch auf realen Fahrzeugen zu testen. Als Zwischenschritt vom Video zum realen Auto können Modellfahrzeuge verwendet werden, die mit ähnlicher Hardware ausgestattet sind, wie richtige Fahrzeuge, durch ihre verringerte Größe aber kostengünstiger sind. Die Entwicklung eines solchen autonomen Modellfahrzeugs ist ein Ziel dieser Arbeit und wird im folgenden Kapitel dargestellt.

3 Modellfahrzeugkonstruktion

Nachdem im vorangegangenen Kapitel 2 die theoretischen Grundlagen zum autonomen Fahren und der digitalen Bildverarbeitung erläutert wurden, kann nun mit der Realisierung der Zielsetzungen dieser Arbeit begonnen werden. Dazu gehört zunächst die Konstruktion eines Modellfahrzeugs, das in der Lage ist, eigenständig zu fahren. Darum soll es in den folgenden Abschnitten gehen.

Zunächst soll dargestellt werden, warum die Konstruktion eines autonomen Modellautos für die HTWK Leipzig sinnvoll ist. Im Anschluss werden die Anforderungen an ein solches Fahrzeug definiert und abschließend gezeigt, wie diese Anforderungen umgesetzt wurden.

Die Konstruktion des hier vorgestellten Modellautos erfolgte durch eine Gruppe von Studenten der HTWK Leipzig im Rahmen des Oberseminars „Autonomes Fahren“. Da der Autor dieser Arbeit vor allem die Planung und Organisation der Entwicklung übernommen hat, soll darauf im Folgenden das Hauptaugenmerk gelegt werden.

3.1 Motivation

Das Team Smart-Driving, welches wie in Kapitel 2.1.2 beschrieben die studentische Forschungsgruppe zum autonomen Fahren an der HTWK Leipzig ist, wurde im Jahr 2014 gegründet. Zu dieser Zeit fand erstmalig der AADC statt, welcher ein von Audi veranstalteter Wettbewerb zum autonomen Fahren ist. Nach erfolgreicher Bewerbung erhalten zehn Studentengruppen aus Deutschland, Österreich oder der Schweiz von Audi ein Modellfahrzeug, welches mit Sensorik und einem Computer ausgestattet ist. Die Teams haben dann ein halbes Jahr Zeit, Software für diese Modellfahrzeuge zu entwickeln, damit sich die Autos selbstständig über unbekannte Miniaturstraßen bewegen können.

Die bisherige Forschungsarbeit des Team HTWK Smart-Driving beschränkte sich auf die Teilnahme am AADC 2014/15 und 2015/16. Für den Zeitraum des Wettbewerbs von September bis März fand sich ein fünfköpfiges Team aus Studenten zusammen und arbeitete intensiv an der Umsetzung der von Audi gestellten Wettbewerbsanforderungen. Da die Modellfahrzeuge nach dem AADC-Finale im März wieder an den Veranstalter übergeben werden mussten, fehlte dem Team HTWK Smart-Driving in den Sommermonaten ein Testobjekt für die Forschung und die Arbeiten wurden weitestgehend eingestellt.

Um dies zu verhindern, entstand an der Fakultät für Informatik, Mathematik und Naturwissenschaften (IMN) der HTWK Leipzig die Idee, ein eigenes Modellfahrzeug zu bauen. Dies würde es dem Team HTWK Smart-Driving einerseits ermöglichen, ganzjährig und unabhängig von Audis Wettbewerb Forschung auf dem Gebiet des autonomen Fahrens zu betreiben. Andererseits könnte die Forschungsgruppe mit einem eigenen Fahrzeug auch an anderen Wettbewerben für selbstfahrende Modellautos teilnehmen. Als Beispiel sei hier der Carolo-Cup[1] genannt.

Im Zuge dieser Arbeit soll nun eine erste, prototypische Version eines autonomen Modellfahrzeugs für die HTWK Leipzig entwickelt und gebaut werden. Welche Anforderungen an ein solches Auto gestellt werden, zeigt das nächste Kapitel.

3.2 Anforderungen

Um die Anforderungen an ein autonomes Modellfahrzeug herauszufinden, lohnt sich ein Blick auf das Fahrzeug aus dem AADC 2015/16. Dieses kann, wie im vorangegangenen Abschnitt erwähnt, als Inspiration für das im Zuge dieser Arbeit konstruierte Fahrzeug bezeichnet werden. Abbildung 3 zeigt das Fahrzeug aus dem AADC 2015/16, welches Audi den teilnehmenden Teams zu Verfügung gestellt hat. Abbildung 4 stellt das detaillierte Hardwarelayout dieses Modellautos dar, auf welches nachfolgend näher eingegangen werden soll.

[1] https://wiki.ifr.ing.tu-bs.de/carolocup/carolo-cup

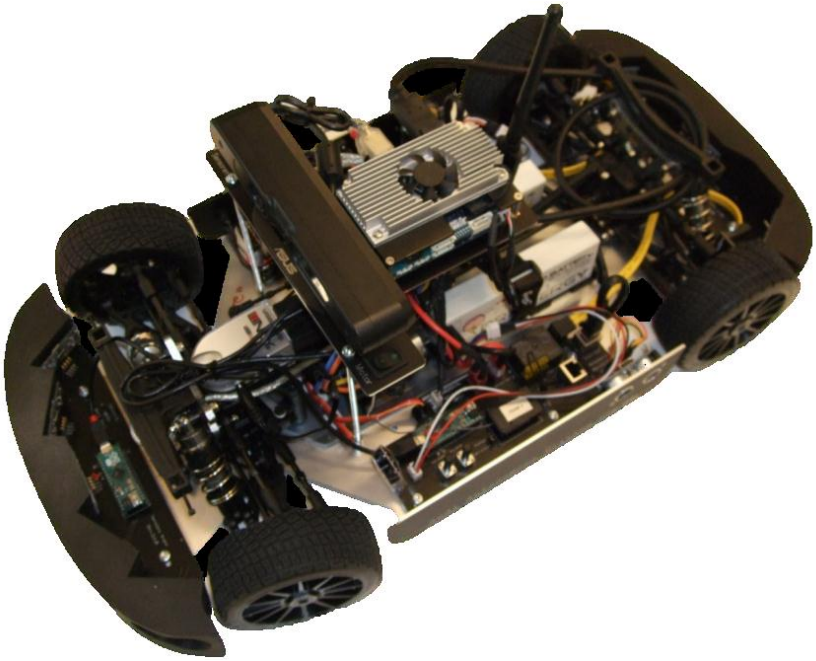

Abbildung 3 – Modellfahrzeug des AADC 2015/16

(entnommen aus [Au15])

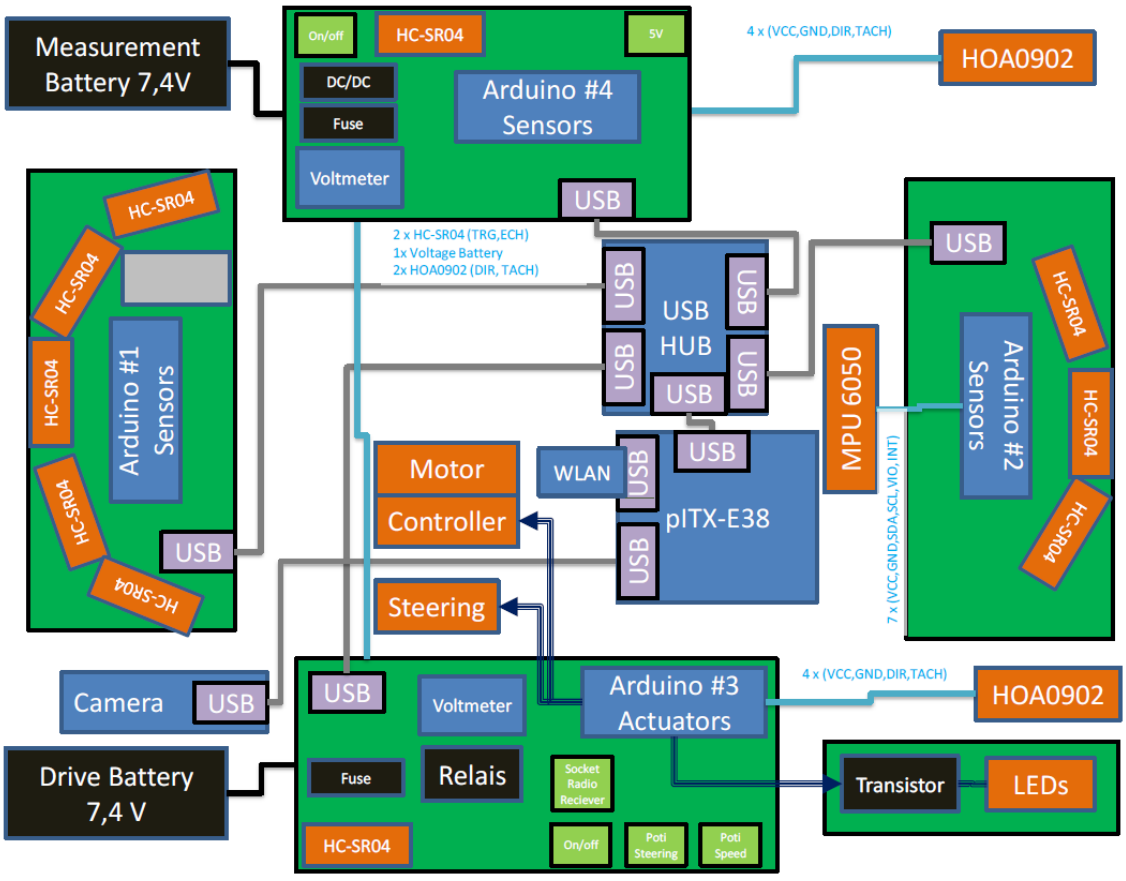

Abbildung 4 – Hardwarelayout des Modellfahrzeugs des AADC 2015/16

(entnommen aus [Au15])

Das Modellfahrzeug von Audi basierte auf einem fernsteuerbaren Auto im Maßstab 1:8, welches um verschiedene Sensoren und Rechentechnik erweitert wurde. Als Hauptrechner wurde ein „Embedded-Mainboard“ mit festverlötetem Prozessor, Grafikeinheit und Arbeitsspeicher verbaut. Dieses war über die Universal-Serial-Bus-Schnittstelle (USB) mit vier Mikrocontrollern verbunden, welche die hardwarenahe Kommunikation mit der Sensorik bzw. Aktorik realisierten. Als Sensorik kam neben zehn Ultraschallsensoren eine Farb- und Tiefenbildkamera, ein Gyroskop und Beschleunigungsmesser sowie ein Raddrehzahlmesser zum Einsatz. Die Aktorik im Sinne von Motor und Lenkung wurde aus dem fernsteuerbaren Auto übernommen und durch eine Haube ergänzt, die ansteuerbare Fahrzeuglichter, wie bspw. Blinker, enthielt.

Ausgehend vom Audi Modellfahrzeug als Vorbild wurden für das autonome HTWK-Auto folgende Anforderungen festgelegt, welche erfüllt werden müssen:

- **Fahrgestell**

  Das Fahrgestell soll im Gegensatz zum Audi-Modellfahrzeug nur einen Maßstab von 1:10 besitzen, um einer Wettbewerbsanforderung des Carolo-Cups zu genügen.

- **Motor und Lenkung**

  Damit sich das HTWK-Modellauto fortbewegen kann, werden ein Elektromotor und eine Lenkung zusätzlich zum Fahrgestell gefordert. Diese müssen hardwarenah ansprechbar sein, um eine Steuerung durch die künstliche Intelligenz (KI) zu ermöglichen.

- **Hinderniserkennung**

  Das Fahrzeug muss in der Lage sein, Umgebungsobjekte zu erkennen. Dadurch kann es Hindernissen ausweichen bzw. Kollisionen vermeiden.

- **Sehfähigkeit**

  Das Modellauto muss seine Umwelt visuell wahrnehmen können, um bspw. Fahrbahnmarkierungen folgen zu können.

- **Geschwindigkeits- und Lagebestimmung**

  Um ein sauberes Fahrverhalten zu erreichen, muss das Fahrzeug in der Lage sein, seine aktuelle Geschwindigkeit zu bestimmen. Weiterführend soll es über Sensoren verfügen, die Informationen über die Lage des Autos im Raum bestimmen können.

- **Fahrverhalten**

  Das Fahrzeug soll in der Lage sein, eine in Meter pro Sekunde angegebene Geschwindigkeit unabhängig vom Akku-Ladezustand fahren zu können. Außerdem soll es einen vorgegebenen Kurvenzug abfahren können.

- **Rechentechnik**

  Als zentrale Steuereinheit soll ein Computer im Modellfahrzeug verbaut sein, der sämtliche Sensordaten erhält, die Berechnungen für die KI durchführt und deren getroffene Entscheidungen an die Aktorik weiterleitet.

- **Stromversorgung**

  Da sich das autonome Auto selbstständig fortbewegen soll, muss es eine vollständig auf Batterien oder Akkus basierende Stromversorgung besitzen. Diese muss sowohl die Aktorik, als auch die Rechentechnik und die Sensorik mit Strom versorgen können.

Des Weiteren wurden für das autonome Auto der HTWK folgende Zusatzanforderungen festgelegt, welche nicht zwingend erfüllt werden müssen:

- **Stromversorgung durch Netzteil**

  Neben der mobilen Stromversorgung soll ein Anschluss für ein Netzteil am Fahrzeug zur Verfügung stehen. Sobald ein Netzteil angeschlossen wird, sollen die Batterien oder Akkus vom Stromkreis getrennt werden.

- **Einsatz einer Fernsteuerung**

  Das Fahrzeug soll über eine Fernsteuerung bewegt werden können, um die Arbeit auf einem großen Parcours zu erleichtern. Sobald eine Fernsteuerung mit dem Modellauto verbunden ist, sollen nur die Befehle der Fernsteuerung an die Aktorik weitergeleitet werden.

- **Lichtanlage**

  Um ein korrektes Straßenverhalten abbilden zu können, soll das Fahrzeug eine Lichtanlage besitzen. Diese umfasst die Scheinwerfer, Blinker, Bremslichter und Rückfahrlichter.

- **Ladestromkreis**

  An den oben erwähnten Netzteilanschluss soll ein Ladestromkreis angeschlossen werden, der die Batterien oder Akkus lädt, solange das Netzteil verwendet wird.

- **Optisch ansprechendes Aussehen**

  Das Fahrzeug soll ein Chassis erhalten, das dem Modellauto äußerlich das Aussehen eines realen PKW oder LKW verleiht.

3.3 Umsetzung der Anforderungen

Im vorangegangenen Kapitel wurden die Anforderungen an das autonome Modellfahrzeug für das Team HTWK Smart-Driving definiert. Im folgenden Abschnitt wird auf die Umsetzung der einzelnen Anforderungspunkte eingegangen. Dabei werden technische Details und getroffene Entscheidungen erläutert.

3.3.1 Fahrgestell, Motor und Lenkung

Bei der Auswahl des Fahrgestells wurde gefordert, dass der Maßstab des Modellfahrzeugs 1:10 betragen soll. In diesem Größensegment gibt es eine Vielzahl verschiedener Fahrgestelle.

Grundsätzlich können ferngesteuerte Modellautos in allen Maßstäben auf zwei verschiedene Arten erworben werden. Auf dem Markt existieren zum einen fertige Gesamtpakete, die ein Fahrgestell beinhalten, auf dem ein Motor und eine Lenkregelung verbaut ist. Diese sind mit einem Fernsteuerungsempfänger verbunden, dessen passender Sender meist mitgeliefert wird. Der Vorteil solcher Gesamtpakete ist, dass alle verbauten Teile zusammenpassen und gemeinsam funktionieren. Außerdem wird Zeit gespart, da kein mechanischer Zusammenbau nötig ist.

Zum anderen können die Teile eines Modellautos auch einzeln gekauft werden. Dabei existiert eine größere Auswahl an einzelnen Komponenten als bei den Komplettpaketen. Außerdem können hochwertigere oder besser auf die Anforderungen passende Einzelteile erworben werden. Um ein Modellauto allerdings komplett selbst zusammenzustellen, werden Erfahrungen im Bereich des Modellfahrzeugbaus benötigt. Andernfalls besteht die Gefahr, dass nicht kompatible Einzelteile erworben und verbaut werden.

Für das Modellfahrzeug der HTWK Leipzig wurde entschieden einen Komplettsatz aus Fahrgestell, Motor und Lenkung als Grundlage zu verwenden. Dies hatte zwei Gründe: Zum einen existierten in der Studentengruppe, welche das autonome Auto für die HTWK konstruiert hat, keine Erfahrungen auf dem Gebiet des Modellbaus. Zum anderen waren für das erste Fahrzeug noch keine fein abgestimmten Modellbau-Einzelteile erforderlich, da das Zusammenspiel und die Wirkung der einzelnen Komponenten in Bezug auf das autonome Fahren erst getestet werden sollten.

Die Wahl eines Komplettsatzes fiel auf das ‚1:10 EP Touring-Car 4WD RtR 2.4GHz' der Firma Reely. Dieses Paket zeichnet sich durch leicht anzusteuernde Motor- und Lenkreglung aus. Außerdem besitzt es genug Platz für die Aufbauten, welche die Rechentechnik und Sensorik tragen sollen. Abbildung 5 zeigt das , 1:10 EP Touring-Car 4WD RtR 2.4GHz' in Frontalansicht.

Abbildung 5 – 1:10 EP Touring-Car 4WD RtR 2.4GHz

Um sämtliche Sensorik und Rechentechnik am Fahrzeug zu befestigen, wurde das Fahrgestell um Aufbauten erweitert. Als Grundgerüst dienten Aluminium-Profile. Diese lassen sich leicht zuschneiden und zu verschiedenen Formen verbinden. Auf dem Aluminiumgerüst wurde eine Aluminium-Platte befestigt, die ungefähr die Größe des Fahrgestells besitzt. Auf Ober- und Unterseite dieser Platte konnten dann Rechentechnik und Sensorik befestigt werden.

Das vollständige Hardwarelayout des autonomen Modellfahrzeugs der HTWK Leipzig kann in Abbildung 6 betrachtet werden. Mit dem Wort ‚Aufbau‘ ist dabei die zusätzliche Aluminiumplatte gemeint, die an das originale Fahrgestell angebaut wurde. Von dieser wurde sowohl Ober- als auch Unterseite zur Befestigung von Bauteilen verwendet. Grün markiert sind der Computer und die daran angeschlossene Peripherie bzw. Sensorik. Die Stromzufuhr wurde rot markiert. In jeweils verschiedenen Farben (Orange, Violett und Gelb) wurden die Mikrokontroller und die mit ihnen verbundene Sensorik bzw. Aktorik eingefärbt. Die Bezeichnungen SRF-08, MPU-6050 sowie Minnowboard werden in den kommenden Kapiteln erläutert.

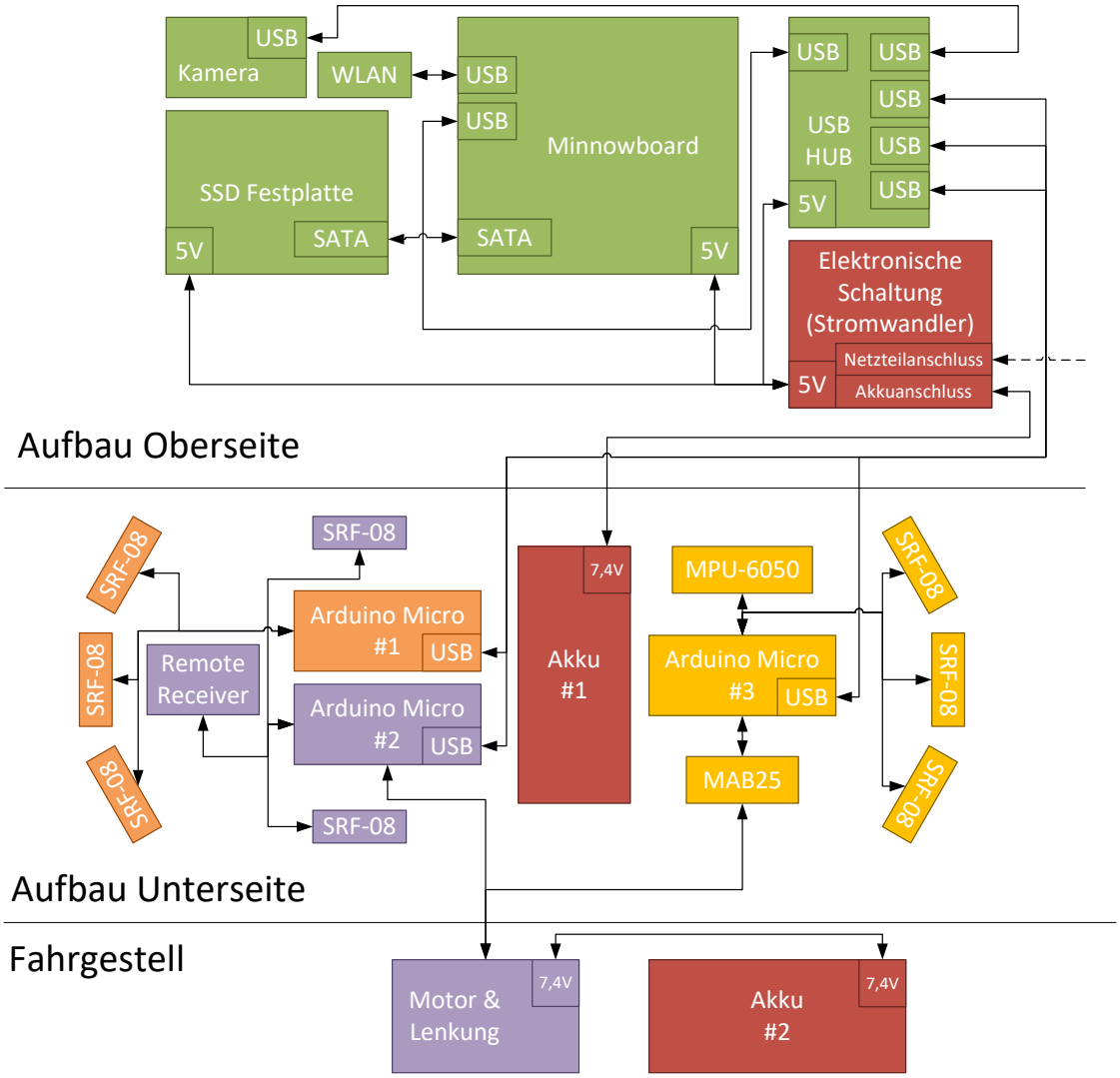

Abbildung 6 – Hardwarelayout des autonomen Modellfahrzeugs der HTWK Leipzig

3.3.2 Hinderniserkennung

Um die geforderte Hinderniserkennung für das autonome Modellfahrzeug umzusetzen, müssen im Auto Sensoren verbaut werden, die die Distanz zu plastischen Objekten im Raum bestimmen können. Das autonome Modellfahrzeug von Audi, welches in dieser Arbeit als Vorbild diente, hatte zur Hinderniserkennung Ultraschall- und Infrarotsensoren sowie eine Tiefenbildkamera verbaut.

Bei der Tätigkeit des Teams HTWK Smart-Driving im AADC 2014/15 und AADC 2015/16 stellte sich heraus, dass sowohl die Tiefenbildkamera als auch die Infrarot-Sensoren unzuverlässig arbeiteten. Die Tiefenbildkamera lieferte ein stark verrauschtes Bild, aus welchem zunächst keine zuverlässigen Informationen gewonnen werden konnten. Durch Mittelwertbildung über mehrere Tiefenbilder hinweg konnten nutzbare Informationen aus den Bildern extrahiert werden. Im Austausch dafür konnten die Grenzen von Hindernissen nun nicht mehr bestimmt werden, da diese über die Mittelwertbildung verwischt wurden. Die Infrarotsensoren waren durch ungenaue bzw. falsche Messungen bereits im AADC 2014/15 aufgefallen und von Audi im AADC 2015/16 entfernt worden.

Einzig die Ultraschallsensoren konnten mit dauerhaft korrekten Messungen in den letzten zwei Jahren überzeugen. Aus diesem Grund wurde für das HTWK Modellfahrzeug entschieden, nur Ultraschallsensoren zur Hinderniserkennung zu verwenden. Audi verwendet an seinem Modellfahrzeug den Sensor ‚HC-SR04', am HTWK Modellauto wurde hingegen der deutlich teurere Sensor ‚SRF08' verbaut. Der ‚SRF08' löst seine Messungen zwar nur in ganzen Zentimetern auf, während der ‚HC-SR04' seine Ergebnisse theoretisch auf 0,3 cm genau auflöst. Dafür sind die Messungen des ‚SRF08' im direkten Vergleich auf 1,5 cm genau, während der ‚HC-SR04' eine Ungenauigkeit von im Schnitt 6,7 cm aufwies. Der schematische Versuchsaufbau zur Bestimmung dieser Werte wird in Abbildung 7 dargestellt, die Ergebnisse dieses Versuchs in Tabelle 1.

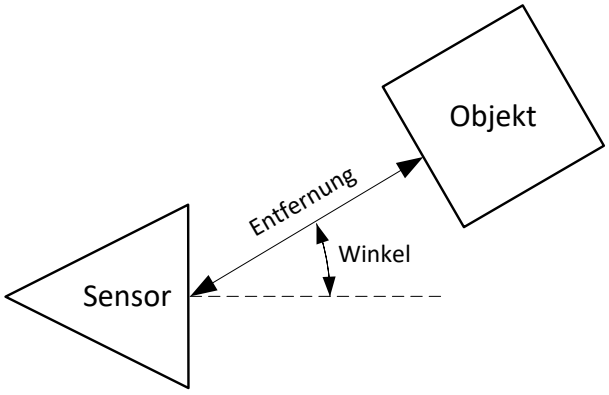

Abbildung 7 – Schematischer Versuchsaufbau zur Bestimmung der Genauigkeit der Ultraschallsensoren ‚HC-SR04' und ‚SRF08'

Tabelle 1 – Versuchsergebnisse bei der Bestimmung der Genauigkeit der Ultraschallsensoren ‚HC-SR04' und ‚SRF08'

		Durchschnittliches Messergebnis nach 10 Messungen in Meter	
		HC-SR04	**SRF-08**
Objektentfernung zum Sensor in Meter	**Winkel zum Sensor in Grad**		
0,1	**0**	0,14	0,09
	10	0,15	0,1
	20	0,15	0,9
	30	0,16	0,11
0,5	**0**	0,57	0,49
	10	0,55	0,48
	20	0,56	0,5
	30	0,53	0,51
1	**0**	1,1	1,0
	10	1,07	1,02
	20	1,06	0,99
	30	Nicht erkannt	1,03
2	**0**	2,08	2,02
	10	2,15	2,03
	20	Nicht erkannt	1,98
	30	Nicht erkannt	2,04

Aus den Versuchsergebnissen ist ebenfalls abzulesen, dass der ‚SRF-08' auch bei größeren Entfernungen und steileren Winkeln Objekte noch genau erkennt. Der ‚HC-SR04' konnte an diesen Stellen keine Hindernisse mehr wahrnehmen. Ein weiterer Vorteil des ‚SRF-08' gegenüber dem ‚HC-SR04' ist, dass er den ansteuernden Mikrocontroller während der Messung nicht blockiert. Die Messung wird vollständig auf dem Chip des ‚SRF-08' durchgeführt, der Mirco-Controller kann sich während dieser Zeit anderen Aufgaben widmen. Aufgrund der Versuchsergebnisse und der eben genannten Entlastung des Mikrocontrollers wurden am autonomen Modellfahrzeug der HTWK insgesamt acht ‚SRF-08' Sensoren verbaut.

3.3.3 Sehfähigkeit

Nachdem das autonome Modellfahrzeug durch die Ultraschallsensoren die Möglichkeit besitzt, Hindernisse zu erkennen, benötigt es nun die Fähigkeit, auf Reize zu reagieren, die nur visuell erkannt werden können. Dazu zählen bspw. Fahrbahnmarkierungen, Schilder oder Ampelanlagen. Damit das autonome Modellfahrzeug seine Sehfähigkeit erlangen kann, benötigt es also eine Videokamera.

Beim Vorbildfahrzeug von Audi wurde eine ‚ASUS XTion'-Kamera verwendet. Diese vereint eine Video- und eine Tiefenbildkamera. Die Schwächen der Tiefenbildkamera wurden bereits in Abschnitt 3.3.2 ausführlich besprochen. Am Videobild der ‚ASUS XTion'-Kamera störte im AADC 2014/15 und 2015/16 vor allem der geringe Sichtwinkel. Dies sorgte in Kurven dafür, dass die äußeren Spurmarkierungen vom Videobild verschwanden. Ebenso konnten Kreuzungen ab eineinhalb Meter vor Kreuzungsbeginn nicht mehr eingesehen werden, was eine Prüfung der Vorfahrtssituation erheblich erschwerte. Aus diesen Grund wurde entschieden, für das HTWK Modellfahrzeug zwei neue Kamerasysteme zu testen. Diese sind zum einen eine Weitwinkelkamera und zum anderen eine Fish-Eye-Kamera. Letztere verspricht einen extremen Sichtwinkel von 90° nach links und rechts. Sie hat allerdings den Nachteil, dass durch ihre Speziallinse Linien, welche in der Realität gerade sind, im Bild gekrümmt dargestellt werden. Zum Vergleich der Sichtwinkel zeigen Abbildung 8 bis Abbildung 10 die gleiche Szene aufgenommen mit den drei erwähnten Kamerasystemen.

Abbildung 8 – Kreuzungsszene aufgenommen mit der ‚ASUS XTion'-Kamera

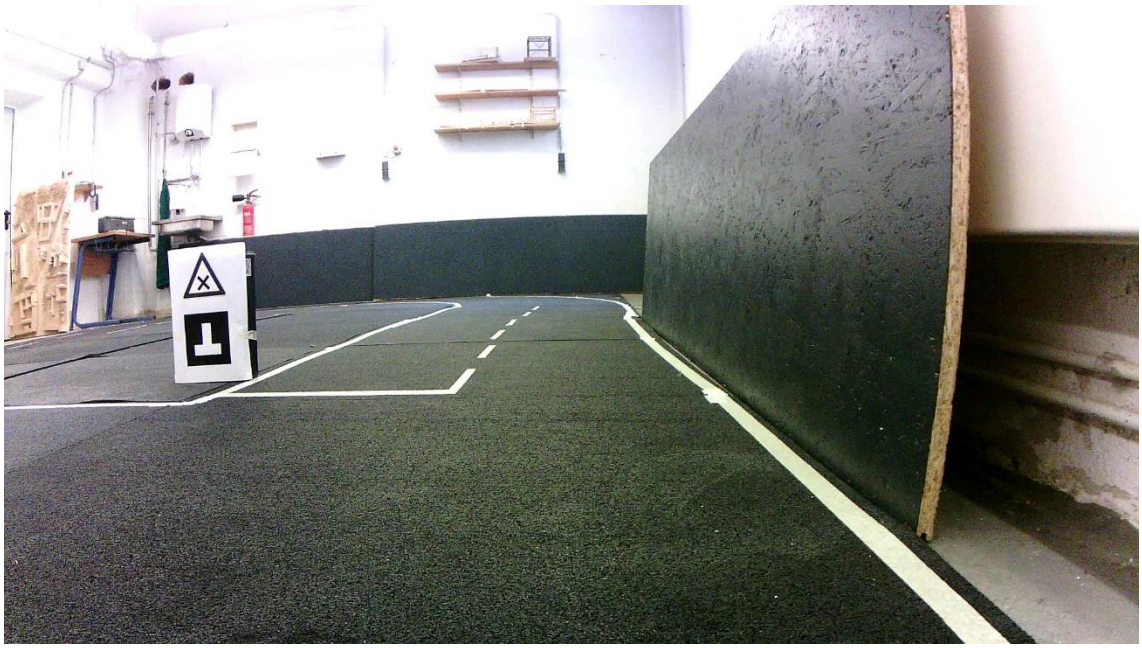

Abbildung 9 – Kreuzungsszene aufgenommen mit der Weitwinkel-Kamera

Abbildung 10 – Kreuzungsszene aufgenommen mit der Fish-Eye-Kamera

In den Abbildungen ist deutlich die Verbesserung der Bildqualität bei der neuen Fish-Eye- und Weitwinkelkamera im Vergleich zur XTion-Kamera zu erkennen. Außerdem zeigen die Bilder, wie die Sichtwinkel von der Xtion- über die Weitwinkel- bis zur Fish-Eye-Kamera ansteigen. Ein erhöhter Sichtwinkel erleichtert auch die Entwicklung der Spurerkennungsalgorithmen, welche in Kapitel 4 behandelt wird. Ob sich die Weitwinkel- oder die Fish-Eye-Kamera besser für die Spurerkennung eignet, wird in Kapitel 4.5.1 genauer thematisiert.

3.3.4 Geschwindigkeits- und Lagebestimmung, Fahrverhalten

Damit sich das Modellfahrzeug autonom bewegen kann, muss es in der Lage sein, eine vorgegebene Geschwindigkeit zu halten. Außerdem soll es Kurvenzüge fahren können. Im einfachsten Fall ist ein Kurvenzug eine Straßenkurve. Es gibt aber auch komplexere Lenksituationen, in denen das Fahrzeug bspw. einer S-Kurve folgen oder zum Überholen einen Spurwechsel vollziehen muss.

Um diese Anforderungen an das Fahrverhalten erfüllen zu können, benötigt das Fahrzeug zunächst Sensoren, mit denen es seine Geschwindigkeit und seine Lage im Raum bestimmen kann. Das Vorbildfahrzeug von Audi verwendete zu diesem Zweck den Inkrementalgeber ‚HOA0902-11' und die inertiale Messeinheit ‚MPU-6050' (engl. inertial measurement unit, IMU).

Der Inkrementalgeber ist ein optischer Sensor, durch den sich eine Lochscheibe dreht und der bei jedem Loch einen Impuls auslöst. Die zeitliche Dauer der Impulse gibt die Rotationsgeschwindigkeit der Drehscheibe an (vgl. [HLN12]). Diese Drehscheibe war beim Audi Modellfahrzeug an der Hinterachse montiert. Somit konnte durch diesen Sensor die Drehgeschwindigkeit der Räder bzw. die Bewegungsgeschwindigkeit des Fahrzeugs gemessen werden.

Die IMU bestimmt dahingegen die Beschleunigung und Orientierung des Fahrzeugs zur Erde. Um die Beschleunigungen auf der Roll-, Nick- bzw. Gierachse zu bestimmen, kommt als Teil der IMU ein Kreiselinstrument (auch Gyroskop genannt) zum Einsatz. In modernen IMUs werden allerdings keine echten Kreisel zur Bestimmung der Kräfte eingesetzt. Die Auslenkung wird bei diesen sogenannten mikro-elektro-mechanischen Systemen (MEMS) über eine schwingende Siliziummasse bestimmt (vgl. [HLN12]).

Bei der Arbeit des Teams HTWK Smart-Driving stellte sich heraus, dass der Inkrementalgeber ‚HOA0902-11' am Audi Fahrzeug insbesondere bei geringen Geschwindigkeiten Fehlimpulse produzierte und somit eine exakte Geschwindigkeitsberechnung unmöglich war. Aus diesem Grund wurde entschieden, am autonomen Auto der HTWK den Hall-Sensor ‚MAB25' zur Drehzahlbestimmung einzusetzen.

Dieser kann die Drehung einer Achse mithilfe des physikalischen Hall-Effekts bestimmen. Dabei rotieren Magneten an stromdurchflossenen Leitern vorbei, wodurch ein Spannungsunterschied im leitenden Material entsteht, welcher gemessen werden kann. Dadurch kann die aktuelle Position der Magneten und demnach der Drehachse bestimmt werden. Der Hall-Sensor hat, im Gegensatz zum Inkrementalgeber, keine Schwäche bei langsamen Drehungen und löst eine 360° Drehung auf 4096 statt nur 32 Abschnitte genau auf.

Der Hall-Sensor greift über ein Zahnrad das Antriebsrad des Modellmotors ab und gibt regelmäßig dessen Position zurück. Dadurch kann die Drehgeschwindigkeit des Antriebs und der Räder bestimmt werden. Ist die Drehgeschwindigkeit der Räder bekannt, kann auch die Fahrzeuggeschwindigkeit berechnet werden.

Eine exakte Beschreibung der Geschwindigkeits- und Lenkregelung am HTWK Modellfahrzeug kann in der weiterführenden Bachelorarbeit von [Fr16] nachgelesen werden. In dieser Arbeit wird detailliert die Umsetzung der Anforderung ‚Fahrverhalten' beschrieben, also das Halten der Geschwindigkeit bzw. Fahren von Kurvenzügen.

3.3.5 Rechentechnik

Um die Anforderungen an die Rechentechnik zu erfüllen, benötigt das autonome Modellfahrzeug der HTWK einen eigenen Bordcomputer. Dieser muss einerseits in der Lage sein, alle Berechnungen der KI des Teams HTWK Smart-Driving durchzuführen. Andererseits muss eine hardwarenahe Anschlussmöglich für die Sensorik und Aktorik am Computer vorhanden sein. Ohne diesen Anschluss könnte die KI keine Sensordaten auswerten und weder Lenkung noch Motor ansteuern.

Bei der Wahl eines Bordcomputers für das autonome Fahrzeug mussten zunächst die Größenbeschränkungen beachtet werden. Auf dem Aufbau des Autos existiert nur ein Platz von 12×12 cm, was die Auswahl an Computern begrenzt. Gerade im Bereich der kleinen Computersysteme existiert eine große Auswahl an Mainboards, welche auf der ARM-Architektur basieren. ARM-Computer zeichnen sich durch eine ‚Reduced Instruction Set Computer'-Architektur (RISC) aus. Das bedeutet, dass der Befehlssatz aus wenigen Befehlen besteht, welche für verschiedene, allgemeine Zwecke eingesetzt werden können (vgl. [Se01]). Dadurch soll es möglich sein, platz- und stromeffiziente Prozessoren zu bauen, welche sich aus diesem Grund besonders für batteriebetriebene, kleine Geräte wie Smartphones, Tablets oder autonome Modellfahrzeuge eignen.

Um die Programmierung einer KI für das Modellfahrzeug zu ermöglichen, wird softwareseitig ein Framework benötigt, das Mittel für die Hardwareabstraktion, modulare Softwareentwicklung sowie Kommunikation zur Verfügung stellt. Nur mithilfe eines solchen Frameworks ist die Entwicklung einer leicht erweiter- und wartbaren Roboter-KI möglich. Audi nutzt zu diesem Zweck das ‚Automotive Data and Time Triggered Framework' (ADTF). Dieses ist aufgrund seines geschlossenen, kommerziellen Lizensierungsmodells allerdings nicht für die Forschung und Lehre an einer Hochschule geeignet. Deshalb wurde für das HTWK Modellfahrzeug beschlossen, das ‚Robot Operating System' (ROS) zu verwenden. Dieses ist unter der Open-Source BSD-Lizenz veröffentlicht und stellt freie Werkzeuge sowie Bibliotheken für Entwickler von Robotik-Systemen zur Verfügung.

Sowohl ROS als auch ADTF waren zum Zeitpunkt der Auswahl der Rechentechnik im April 2016 nur eingeschränkt bzw. gar nicht kompatibel für Computer mit ARM-Architektur verfügbar. Aus diesem Grund wurde beschlossen, einen Computer in der für Desktop- und Server-Computer üblichen x86-Architektur im Fahrzeug zu verbauen.

Die x86-Architektur unterscheidet sich von der ARM-Architektur darin, dass sie eine ‚Complex Instruction Set Computer'-Architektur (CISC) ist. Im Vergleich zur RISC-Architektur sind die Maschinenbefehle der CISC-Architektur abstrakter und mächtiger. Um einen CISC-Befehl im RISC-Befehlssatz nachzustellen, werden meist mehrere RISC-Befehle benötigt. Im Allgemeinen sollen x86-Prozessoren einen höheren Stromverbrauch und eine stärkere Wärmeabgabe als ARM-Prozessoren haben (vgl. [UHF13]).

Für das HTWK Modellfahrzeug wurde schließlich das ‚Minnowboard MAX' als Bordcomputer gewählt. Es nutzt, wie zu diesem Zeitpunkt gefordert, die x86-Architektur. Außerdem zeichnet es sich durch seine kompakte Bauweise mit festverlötetem ‚Intel-Atom'-Prozessor und zwei Gigabyte Arbeitsspeicher aus. Die im Prozessor enthaltene interne Grafikkarte kann außerdem über die ‚Open Computing Language' (OpenCL) angesprochen werden. OpenCL ermöglicht es, die digitale Bildverarbeitung mit wenig Aufwand auf der Grafikkarte berechnen zu lassen und dadurch den Prozessor zu entlasten. Nähere Informationen zum Einsatz von OpenCL finden sie in Kapitel 4.4.1.

Das ‚Minnowboard MAX' wird über eine ‚Solid-State-Drive'-Festplatte (SSD) mit Speicher versorgt. Für den Zugriff auf den Bordcomputer von außen wird außerdem ein WLAN-USB-Stick eingesetzt.

Um die hardwarenahe Abfrage der Sensorik und Ansteuerung der Aktorik zu ermöglichen, werden wie beim Vorbild von Audi drei ‚Arduino Micro'-Mikrocontroller am HTWK Modellfahrzeug eingesetzt. Diese bieten Möglichkeiten zur hardwarenahen Kommunikation und werden über USB an den Bordcomputer angeschlossen. Mit 5×2 cm sind sie platzsparend, haben aber auch weniger Arbeits- und Programmspeicher als größere Mikrocontroller. Mithilfe der ROS-Bibliothek ‚ROS-Serial' konnte eine leicht erweiterbare Kommunikation zwischen dem Bordcomputer und den Mikrocontrollern programmiert werden.

3.3.6 Stromversorgung

Damit sich das Modellfahrzeug autonom bewegen kann, muss es über eine mobile Stromversorgung verfügen. Dazu werden am Auto zwei Akkus eingesetzt. Einer versorgt den Motor und die Lenkung mit Strom, der andere betreibt sämtliche zusätzlich ans Fahrgestell angebrachte Elektronik. Genau wie beim Vorbildfahrzeug von Audi wurden am HTWK Auto ‚Brainergy'-Akkus mit 5200 mAh Ladung verbaut. Mit diesen wurde bereits beim AADC gute Erfahrungen gemacht und es bestand kein Grund, auf unbekannte Akkus zu setzen.

Motor und Lenkung können direkt mit den angegebenen Akkus betrieben werden. Zur Versorgung der Bordelektronik muss mit dem Akku allerdings eine eigens entworfene elektronische Schaltung genutzt werden. Diese erfüllt zwei Aufgaben:

- **Anpassen der Betriebsspannung**

  Die ‚Brainergy'-Akkus liefern eine Spannung von 7,4V. Die gesamte Bordelektronik benötigt allerdings nur eine Betriebsspannung von 5V. Die entwickelte elektronische Schaltung reduziert die Spannung der Akkus auf das benötigte Maß.

- **Nutzung eines Netzteils**

  Als optionale Anforderung für das autonome Fahrzeug war angegeben, dass eine Möglichkeit existieren soll, das Auto zu Entwicklungszwecken zwischenzeitlich auch per Netzteil mit Strom zu versorgen. Die elektronische Schaltung ermöglicht einen unterbrechungsfreien Wechsel der Stromversorgungsquelle zwischen Akku und Netzteil.

Abbildung 11 zeigt den Schaltplan der elektronischen Schaltung, welche die beiden soeben genannten Anforderungen erfüllt. Sie wurde von Andrej Lisnitzki und Martin Morcinietz entworfen.

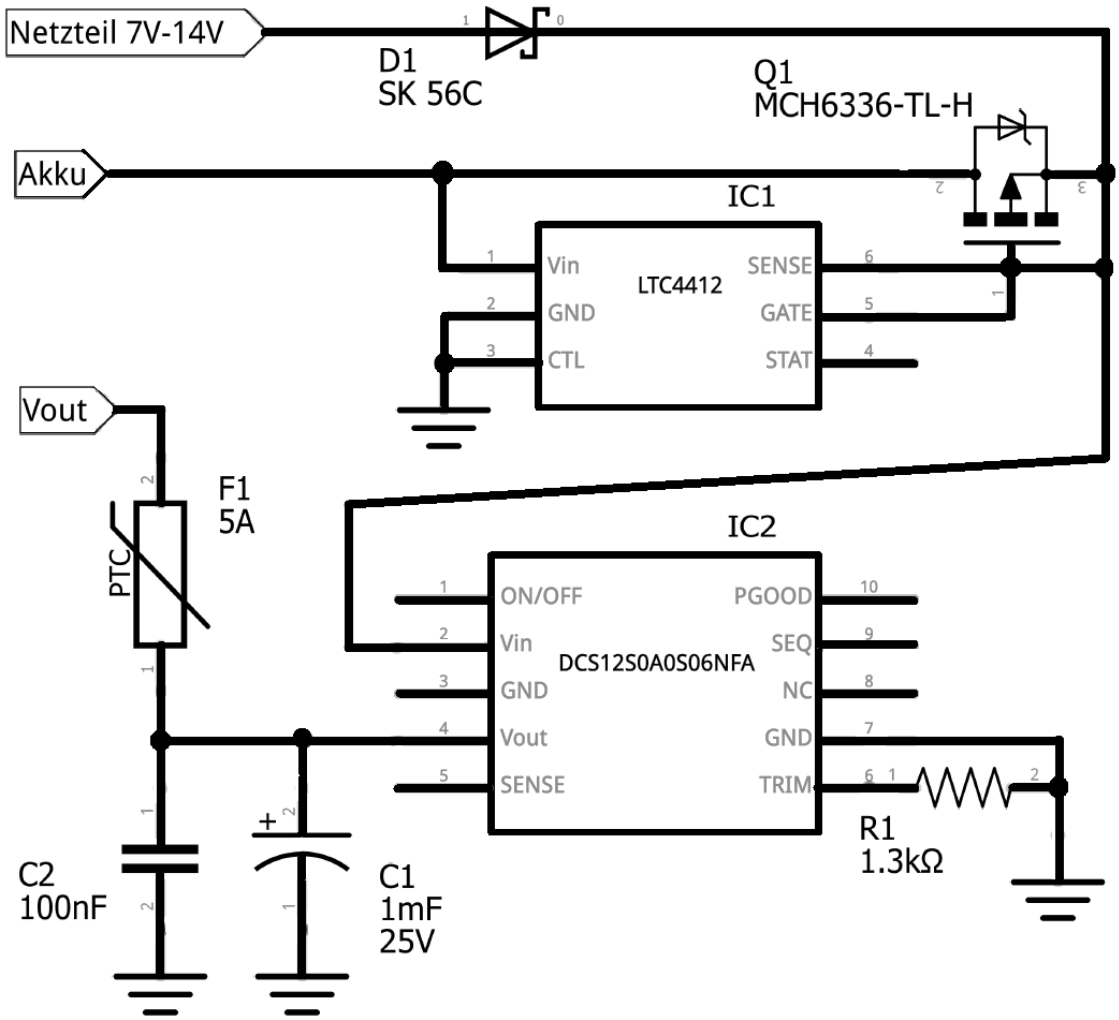

Abbildung 11 – Schaltplan der elektronischen Schaltung für die Stromversorgung des HTWK Modellfahrzeug
(entworfen von Andrej Lisnitzki und Martin Morcinietz)

3.4 Verbesserungsvorschläge

Die Gesamtentwicklungszeit des autonomen Modellfahrzeugs der HTWK betrug sieben Monate von April bis Oktober 2016. Die Entwicklungsarbeit leistete dabei ein Team bestehend aus den Studenten Patrick Bachmann, Kay Szesny, Tino Weidenmüller, Martin Morcinietz und Andrej Lisnitzki unter der Leitung des Studenten Georg Jenschmischek und der Professorin Dr. Sibylle Schwarz. Es wurden alle zwingenden Anforderungen und zwei optionale Anforderungen an das Fahrzeug erfüllt. Das fertige Modellfahrzeug kann in Abbildung 12 betrachtet werden.

Abbildung 12 – HTWK Modellfahrzeug

Obwohl alle zwingenden Anforderungen an das autonome Modellauto der HTWK Leipzig erfüllt wurden, bleibt dieses Fahrzeug ein erster Prototyp seiner Art. Bei seiner Entwicklung konnte das Team viele Erfahrungen sammeln. Im Nachhinein können folgende Verbesserungsvorschläge für das perspektivisch nächste HTWK Modellfahrzeug unterbreitet werden:

- **Erfüllung der optionalen Anforderungen**

  Zu den nicht erfüllten, optionalen Anforderungen zählen die Lichtanlage, der Ladestromkreis und das optische ansprechende Aussehen. Gerade die Lichtanlage sollte beim nächsten autonomen Auto der HTWK von Anfang an mit geplant werden, da sie ein wichtiger Bestandteil sowohl des AADC, als auch des Carolo-Cups ist. Außerdem gehört zum vollständig korrekten Verhalten eines autonomen Fahrzeugs auch das Setzen der Lichtsignale, weswegen das Modellfahrzeug eine Möglichkeit dafür bieten sollte.

Ein Ladestromkreis für die Akkus ist schwierig zu implementieren, da für das Aufladen der bisher verwendeten Akkus ein Balancer benötigt wird, der die Spannung der einzelnen Zellen des Akkus angleicht. Eine solche Ladelogik würde viel Entwicklungszeit und Platz auf dem Fahrzeug kosten und sollte deshalb beim nächsten Modellauto nicht als Anforderung gelistet werden.

- **Beachtung der Anforderungen des Carolo-Cups**

  Perspektivisch soll mit dem Team HTWK Smart-Driving und seinem neuen Modellfahrzeug eine Teilnahme am Carolo-Cup angestrebt werden. Dieser Wettbewerb gibt zwei Anforderungen vor, welche für die Entwicklung eines zukünftigen Modellfahrzeugs von Bedeutung sein können: Zum einen sollen Carolo-Cup-Autos möglichst kostengünstig und energieeffizient sein. Zum anderen gilt es beim Carolo-Cup möglichst schnell einen gegebenen Parcours zu durchfahren. Beide Anforderungen wurden bei dem Fahrzeug, welches im Zuge dieser Arbeit entstanden ist, nicht beachtet und sollten für die Zukunft eine Rolle bei der Entwicklung spielen.

- **Verwendung von anderen oder mehr Mikrocontrollern**

  Bei der Entwicklung des Codes, welcher die Mikrocontroller steuert, fiel auf, dass die drei ‚Arduino Micro'-Mikrocontroller insgesamt über zu wenig Arbeitsspeicher für die zu erfüllenden Aufgaben verfügten. Dadurch kam es zu unerwartetem Verhalten der Sensorik bzw. Aktorik. Mit viel Optimierungsaufwand konnte der Code so reduziert werden, dass alle Mikrocontroller korrekt arbeiteten. Allerdings ist dieser Code nun kaum erweiter- oder wartbar. Um dies in Zukunft zu verhindern, sollten beim nächsten Modellfahrzeug entweder stärkere Mikrocontroller in geringerer Stückzahl eingesetzt oder ein zusätzlicher ‚Arduino Micro'-Mikrocontroller verbaut werden.

- **Verwendung eines Bordcomputers mit ARM-Architektur**

  Wie bereits in Kapitel 3.3.5 erwähnt, wurde nur auf einen Bordcomputer mit ARM-Architektur verzichtet, weil ROS zu diesem Zeitpunkt noch nicht für ARM-Betriebssysteme freigegeben war. Dies hat sich inzwischen geändert, sodass die größere Auswahl an ARM-Computern auch für das HTWK Modellfahrzeug nutzbar ist. Die Wahl des Bordcomputers sollte für das nächste autonome Auto erneut getroffen werden.

Beim Praxistest der in dieser Arbeit entwickelten Spurerkennung (vgl. Kapitel 4.5) fiel außerdem auf, dass der Prozessor des Bordcomputers leistungsmäßig zu klein dimensioniert ist, um Sensorverarbeitung und KI berechnen zu können. Der kommende Bordcomputer sollte einen deutlich leistungsstärkeren Prozessor beinhalten.

- **Verbesserte Weiterleitung des Signals der Fernsteuerung**

  Im Augenblick wird das Signal der Fernsteuerung durch einen Mikrocontroller geleitet, sofern die autonome Steuerung deaktiviert wurde. Da das Fernsteuerungssignal über Pulsweitenmodulation codiert und die Abtastrate des Mikrocontrollers zu gering ist, werden die Fernbedienungssignale nicht korrekt durch den Mikrocontroller imitiert. Dadurch springen Motor- und Lenkansteuerung bei der Nutzung der Fernbedienung zwischen verschiedenen Werten hin und her, was zu ungleichmäßigen Bewegungen des Fahrzeugs führt.

  Um dies zu verhindern, sollte beim künftigen Modellfahrzeug eine direkte Verbindung zwischen dem Empfänger der Fernbedienung und dem Motor bzw. der Lenkung existieren. Über einen elektronischen Schalter sollte ein Mikrocontroller in der Lage sein, diese Verbindung nach Bedarf zu verbinden oder zu trennen.

- **Einsatz einer SD-Karte statt einer SSD-Festplatte**

  Die im Fahrzeug verbaute SSD-Festplatte wird nur zur Speicherung des Betriebssystems und der Log-Dateien verwendet. Beide Aufgaben könnten ebenso von einer SD-Karte erfüllt werden, die an den SD-Karten-Steckplatz des Mainboards angeschlossen wird. Dadurch kann der Platz der Festplatte auf dem Oberdeck des Fahrzeugs eingespart werden. Es wird außerdem kein zusätzlicher Stromanschluss für die SD-Karte benötigt, wie es bei der SSD der Fall war.

  Zu prüfen ist, ob die niedrigeren Lese- und Schreibgeschwindigkeiten der SD-Karte für die Zwecke des autonomen Fahrzeugs genügen.

Insgesamt kann das Ziel der Vollendung des ersten autonomen Modellfahrzeugs der HTWK Leipzig als erfüllt betrachtet werden. Im folgenden Kapitel 4 wird nun mit der Bearbeitung der zweiten Zielstellung dieser Arbeit begonnen, indem mit digitaler Bildverarbeitung auf dem konstruierten Modellauto eine Spurerkennung entwickelt wird. In der Bachelorarbeit von [Fr16] wird ebenfalls mit dem HTWK Modellfahrzeug gearbeitet. Darin entwickelt [Fr16] eine Lenk- und Geschwindigkeitsregelung für das autonome Auto.

4 Digitale Bildverarbeitung

Im vorherigen Kapitel wurde gezeigt, wie im Zuge dieser Arbeit ein autonomes Modellfahrzeug konstruiert wurde. Wie im Abschnitt 3.1 erwähnt, war die Möglichkeit einer ganzjährigen Forschung des Teams Smart-Driving die hauptsächliche Motivation für den Bau des Modellautos. Im zweiten Teil dieser Arbeit soll mithilfe digitaler Bildverarbeitung eine erste Forschungsarbeit mit dem neuen Modellfahrzeug erfolgen. Genauer gesagt, geht es darum, die bisherige Spurerkennung aus den AADC-Wettbewerben zu verbessern.

Spurerkennung beschreibt dabei das Entdecken und Verfolgen von Fahrbahnmarkierungen im Kamerabild eines autonomen Fahrzeugs. Dazu werden Methoden der digitalen Bildverarbeitung eingesetzt. Nach dem erfolgreichen Erkennen der Fahrspur muss diese in ein Spurmodell überführt werden. Mithilfe dieses Modells kann im Anschluss eine Lenkregelung am Fahrzeug erfolgen, die selbiges innerhalb der Fahrspur hält. In dieser Arbeit soll nur die Spurerkennung im Kamerabild betrachtet werden.

Wie die bisherige Spurerkennung funktioniert und warum eine Verbesserung überhaupt notwendig ist, wird im folgenden Unterkapitel dargestellt.

4.1 Ausgangssituation

Um die Anforderungen der AADC-Wettbewerbe in den letzten zwei Jahren zu erfüllen, musste das Team HTWK Smart-Driving unter anderem dafür sorgen, dass das Modellfahrzeug autonom innerhalb der vorgegebenen Miniatur-Straßenlandschaft fährt. Die Wettbewerbsjury zog sowohl beim Verlassen der Fahrbahn, als auch beim Schneiden von Fahrbahnmarkierungen Wertungspunkte vom Teamergebnis ab. Alles in allem gilt das Spurhaltesystem als Grundbaustein für sauberes, autonomes Fahren, da alle anderen Fahraufgaben davon abhängen: Aufgrund schlechter Positionierung auf der Fahrbahn können Parklücken nicht angefahren werden oder sie werden gar nicht erst erkannt. Weiterhin können beim Überholen keine Spurwechsel vollzogen werden und es wird das Abbiegen an Kreuzungen erschwert. Aus diesen Gründen flossen bereits zwei Jahre Forschungsarbeit in die Spurerkennung des Teams Smart-Driving. Im Folgendem wird die Funktionsweise der bisher eingesetzten Spurerkennung vorgestellt.

4.1.1 Bisher eingesetzter Spurerkennungsalgorithmus

Der im AADC 2014/15 genutzte Spurerkennungsalgorithmus beruht im Wesentlichen auf der Annahme, dass Spurmarkierungen im IPM-Bild einer Straße immer an den Rändern des IPM-Bildes enden. Die Randpunkte, zwischen denen die Fahrbahnlinien verlaufen, können verfolgt werden. Sollte eine Spurmarkierung unterbrochen sein, kann sie anhand des gespeicherten Randpunktes wiedergefunden werden, sobald sie erneut im Bild erscheint.

Der gesamte Spurerkennungsalgorithmus besteht aus drei Phasen, wobei der eben erwähnte Ansatz die letzte Phase bildet. Zur Vorbereitung der Spurerkennung erfolgen zunächst eine Fluchtpunktbestimmung und ein IPM.

4.1.1.1 Phase 1: Fluchtpunktbestimmung

Der Fluchtpunkt eines Perspektivbildes ist der Punkt, an dem sich alle Linien im Bild schneiden, die in der Realität auf der gleichen Raumebene liegen. Übertragen auf den Straßenverkehr liegen bspw. alle Fahrbahnmarkierungen auf der gleichen Raumebene und schneiden sich deshalb im Bild in einem Fluchtpunkt. In der Realität verlaufen Straßenmarkierungen im Gegensatz dazu parallel und würden sich niemals schneiden. Abbildung 13 zeigt beispielhaft den Fluchtpunkt einer Straßenszene.

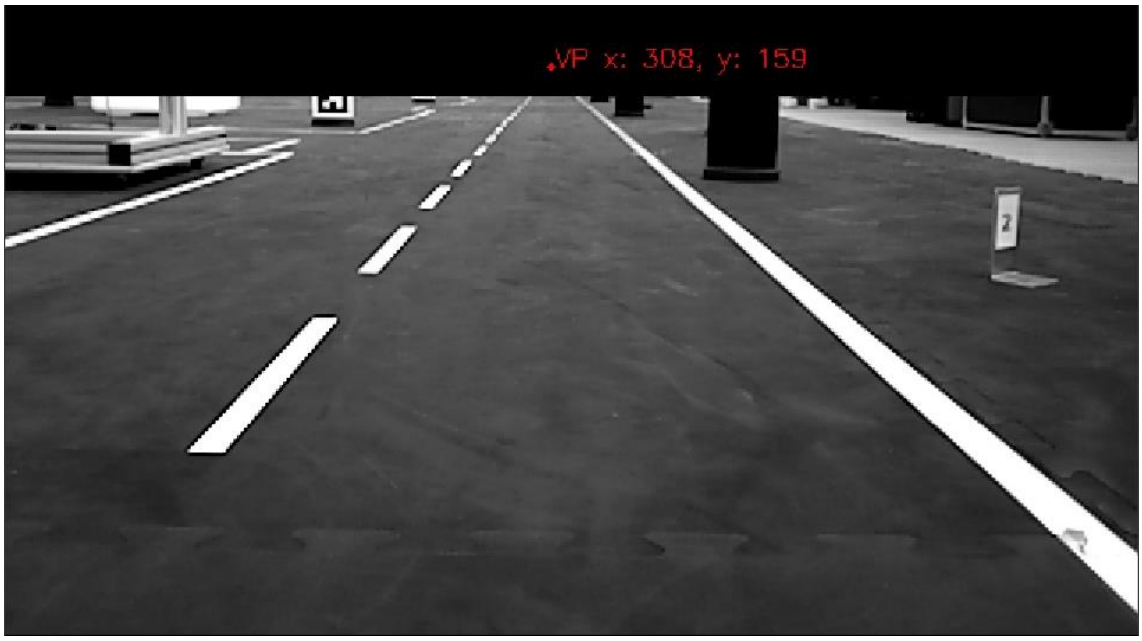

Abbildung 13 – Fluchtpunt (VP) einer Straßenszene

Die Fluchtpunktbestimmung als Teil der bisher verwendeten Spurerkennung basiert auf der von [Hu06] vorgestellten Methode. Zusammengefasst wird zunächst ein Hough-Algorithmus (s. Kapitel 2.2.2.1) auf das Bild angewendet, welcher die Spurmarkierungen als markante Linien erkennt. Wie eben erwähnt, befindet sich der Fluchtpunkt im Schnittpunkt der Fahrbahnmarkierungen. Resultierend daraus wird mittels der Methode der kleinsten Quadrate ein Punkt berechnet, der möglichst nah an allen Hough-Linien liegt.

Dieser Punkt wird in die Berechnung des Fluchtpunktes auf dem nächsten Videobild übertragen, da anzunehmen ist, dass die Varianz der Fluchtpunkt zweier aufeinanderfolgender Videobild geringfügig sein wird. Bei der Auswahl des Punktes, der allen Hough-Linien möglichst nahekommt, kann nun der Abstand zum vorherigen Fluchtpunkt als Bewertungskriterium herangezogen werden.

Dieser Algorithmus wird solange auf den Videobilderstrom angewandt, bis der Fluchtpunkt kaum Änderungen zum Fluchtpunkt des vorhergehenden Bildes aufweist. Dieser Punkt kann als Kamerafluchtpunkt bezüglich der Fahrbahnebene festgehalten werden, da die Kamera während der Fahrt festmontiert ist. Der berechnete Fluchtpunkt wird in Phase 2 zur Erzeugung des IPM-Bildes benötigt.

Abbildung 14 zeigt die gleiche Straßenszene wie Abbildung 13 mit grün markierten Hough-Linien und rot markiertem Fluchtpunkt.

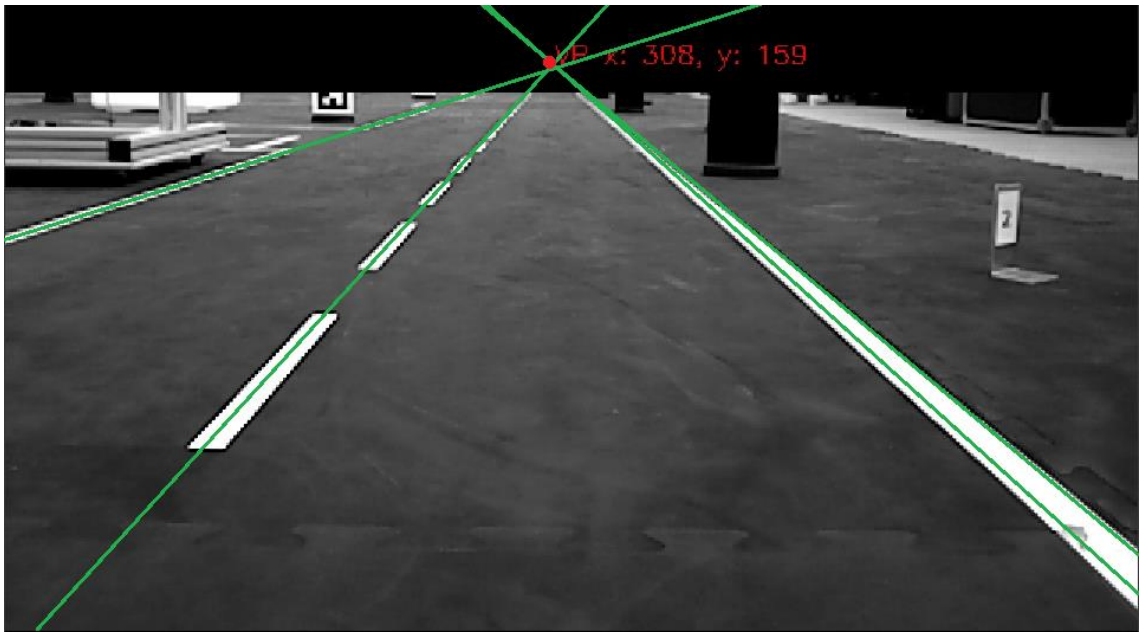

Abbildung 14 – Straßenszene mit Hough-Linien (Grün) und Fluchtpunkt (VP, Rot)

4.1.1.2 Phase 2: IPM-Erzeugung

Die Erzeugung des IPM-Bildes aus dem Videobild dient vor allem dem Zweck, dass in einem korrekt erzeugten IPM-Bild Entfernungen gemessen werden können. Dies bedeutet genauer gesagt, dass ein Bildpunkt im IPM-Bild einer vorher definierten Entfernung in der Realität entspricht. Werden also zwei IPM-Bildpunkte und deren in die Realität abgebildeten Konterpart-Punkte betrachtet, so verhält sich die Strecke zwischen den Bildpunkten direkt proportional gegenüber der Strecke dieser Punkte in der Realität. Dadurch kann die Position des Fahrzeugs auf der Fahrspur unmittelbar aus dem Bild abgelesen werden.

Um ein solches, proportionengetreues IPM-Bild aus dem Videobild zu erzeugen, muss jeder Videobildpunkt auf einen IPM-Bildpunkt abgebildet werden. Diese Abbildungsfunktion f_{IPM} kann mithilfe des Lochkamera-Modells berechnet werden, wie [BBF98] zeigen. Die Abbildungsfunktion f_{IPM} besitzt folgende Parameter:

- Die Verschiebung der Kamera zum Koordinatenursprung im IPM-Bild d_x , d_y . Da der Mittelpunkt der Frontstoßstange als Koordinatenursprung festgelegt wurde, war die Kamera um $d_y = 15cm$ nach hinten sowie um $d_x = 3cm$ nach rechts versetzt.
- Die Höhe der Kamera über dem Boden d_z . Die Kamera im AADC befand sich $d_z = 25cm$ über dem Boden.
- Die Brennweite der Kamera Br . Die Kamera im AADC hatte eine Brennweite von $Br = 525$.
- Die Öffnungswinkel der Kamera horizontal α_h sowie vertikal α_v . Diese betrugen bei der XTion-Kamera im AADC $\alpha_h = 29°$ bzw. $\alpha_v = 23°$.
- Die Bildgröße des Ausgangsbildes horizontal G_h sowie vertikal G_v . Im AADC nutzte das Team Smart-Driving die Bildgröße $G_h = 640$ mal $G_v = 480$

- Den Neigungswinkel der Kamera δ zur Projektionsfläche. Dieser wird anhand des vertikalen Abstands d_{FP} des Fluchtpunkts FP zur Bildmitte ermittelt.

$$d_{FP} = \left| FP_y - \frac{G_h}{2} \right|$$

Mithilfe trigonometrischer Beziehungen im Lochkameramodell kann ausgehend von Abstand d_{FP} und der Brennweite Br dynamisch der Neigungswinkel δ berechnet werden.

$$\delta = \tan^{-1} \frac{\mathrm{d_{FP}}}{Br}$$

Die Abbildungsfunktion f_{IPM} nimmt die x und y-Koordinate eines Bildpunktes entgegen und berechnet die Position dieses Bildpunktes im IPM-Bild.

$$f_{IPM_x}(x,y) = d_z * \cot\left((\delta - \alpha_v) + \frac{2y * \alpha_v}{G_v}\right) * \cos\left(-\alpha_h + \frac{2x * \alpha_h}{G_h}\right) - d_y$$

$$f_{IPM_y}(x,y) = d_z * \cot\left((\delta - \alpha_v) + \frac{2y * \alpha_v}{G_v}\right) * \sin\left(-\alpha_h + \frac{2x * \alpha_h}{G_h}\right) - d_x$$

Wendet man diese Berechnungen auf das Kamerabild aus Abbildung 13 und Abbildung 14 an, wird das IPM-Bild aus Abbildung 15 erzeugt. Die rote Horizontallinie ist bei 100 Pixel Abstand zum unteren Bildrand eingezeichnet, was einem Abstand von 100 cm zum Fahrzeug entsprechen sollte. Da zwei Mittelstreifen einem Weg von einem Meter entsprechen, ist zu erkennen, dass das IPM-Bild tatsächlich verhältnisgetreu zur Realität ist.

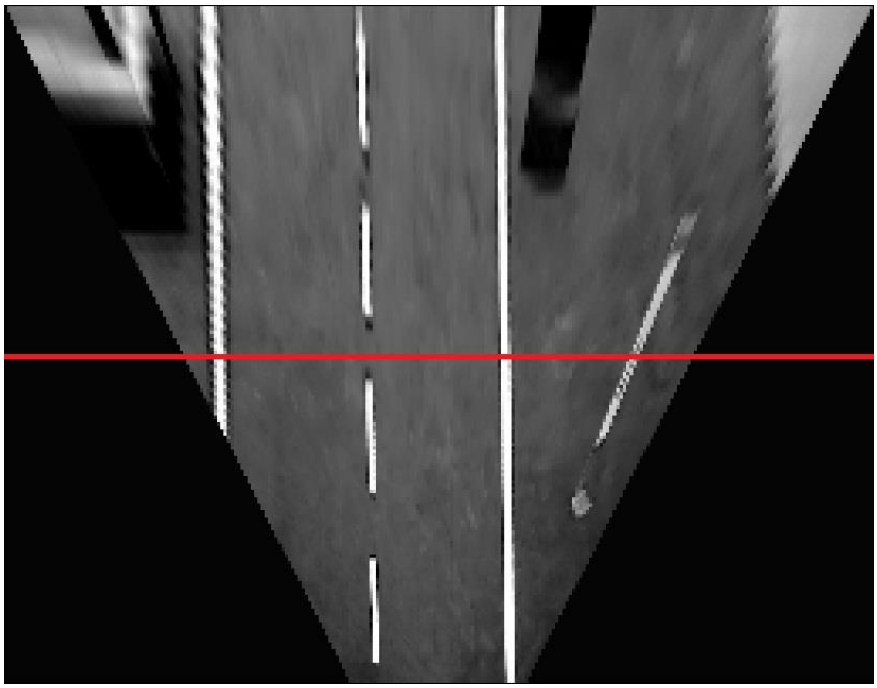

Abbildung 15 – Proportionales IPM-Bild.
Die rote Linie markiert einen 100 Pixel Abstand zum unteren Bildrand bzw. einen 100 cm Abstand zum Fahrzeug

4.1.1.3 Phase 3: Spurrandpunkt-Verfolgung

Nachdem in den zwei vorhergehenden Phasen mithilfe des ermittelten Fluchtpunktes aus dem Video- ein IPM-Bild erzeugt wurde, kann in der dritten Phase die eigentliche Spurerkennung und -verfolgung durchgeführt werden. Dazu werden, wie bereits unter 4.1.1 erwähnt, die Bildpunkte ermittelt und verfolgt, welche zur Spurmarkierung gehören und auf dem Randbereich des IPM-Bildes liegen. Der Randbereich des IPM-Bildes ist dabei nicht gleichbedeutend mit dem Bildrand, da durch die perspektivische Verzerrung in Phase 2 das rechteckige Videobild auf ein trapezförmiges IPM-Bild abgebildet wurde. Abbildung 16 verdeutlicht die soeben genannten Begriffe nochmals. Dort sind der Bildrand blau, der IPM-Rand grün und die Spurrandpunkte der linken Spur rot markiert.

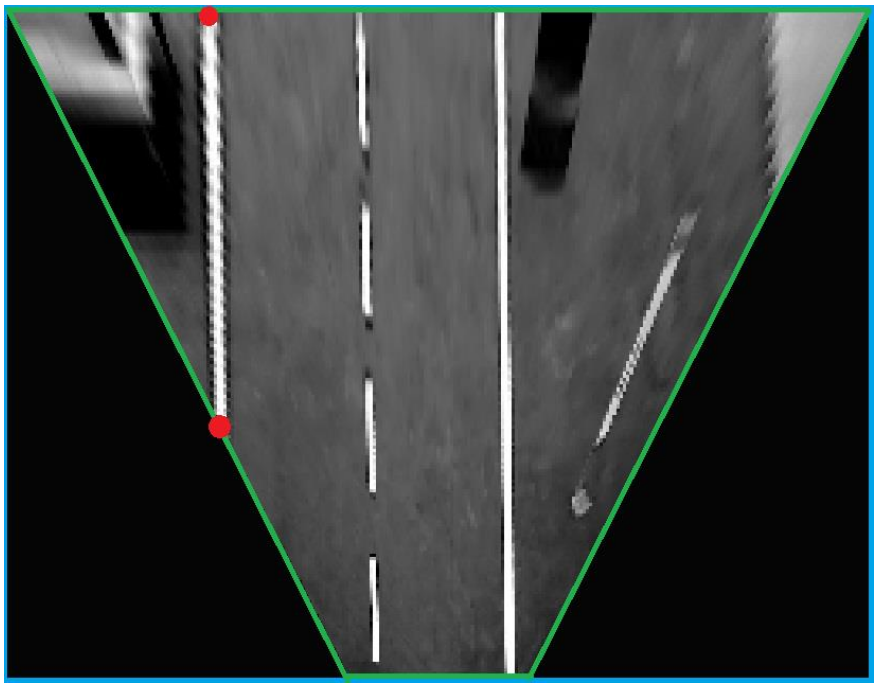

Abbildung 16 – Proportionales IPM-Bild mit Randmarkierungen.
Blau markiert ist der Bildrand, grün markiert ist der IPM-Rand und rot markiert sind die Spurrandpunkte der linken Spur.

Um die Spurrandpunkte zu finden und zu verfolgen, wird das IPM-Bild zunächst binarisiert (vgl. dazu Kapitel 2.2.2.3). Zur Bestimmung des Grenzwertes wird der ‚Yen'-Algorithmus verwendet (vgl. [YCC95]). Dieser erzeugt auch bei dunklen bzw. überbelichteten Bildern einen robusten Grenzwert.

Auf dem binarisierten IPM-Bild wird anschließend ein ‚Connected Component Labeling' durchgeführt. ‚Connected Component Labeling' bezeichnet den Vorgang, dass zusammenhängende, weiße Bereiche des Bildes erkannt und Informationen über diese Bereiche gesammelt werden. Solche Informationen sind bspw. der am weitesten rechts oder oben liegende Punkt, Höhe, Breite und Flächeninhalt.

Zur Initialisierung sucht der Spurerkennungsalgorithmus zwei ‚Connected Components', die rechts bzw. links der Bildmitte liegen und ein möglichst hohes Höhe-Breite-Verhältnis haben. Der Algorithmus geht also davon aus, dass er auf einer Geraden gestartet wird.

Hat der Algorithmus passende ‚Connected Components' gefunden, approximiert er anhand der Punkte der ‚Components' eine Funktion dritten Grades, welche sich dem Spurmarkierungsverlauf annähert. Anschließend werden die Schnittpunkte der Funktion mit den IPM-Rändern bestimmt und gespeichert. Diese Punkte sind die sogenannten Spurrandpunkte.

Im nächsten Videobild wird erneut eine Binarisierung und ein ‚Connected Component Labeling' durchgeführt. Dabei wird ausgehend von den letzten Spurrandpunkten nach ‚Connected Components' gesucht, die zum einen nicht zu weit weg von den letzten Spurrandpunkten liegen und zum anderen keine spurmarkierungstypischen Eigenschaften verletzen. Zu Letzteren zählen neben dem bereits erwähnten hohen Höhe-Breite-Verhältnis ein geringer Flächeninhalt sowie eine maximale Breite. Je nachdem, ob passende ‚Connected Components' gefunden wurden oder nicht, kann die Spur nun einen von vier Zuständen einnehmen. Die neuen Spurrandpunkte werden je nach Zustand auf verschiedene Arten bestimmt. Die vier Zustände sind:

- **Full-Line**

  Der Zustand ‚Full-Line' wird erreicht, wenn in der Nähe beider vorheriger Spurrandpunkte die gleiche ‚Component' gefunden wurde. Die Spurmarkierung ist nicht unterbrochen. Die Spurrandpunkte werden zur gemeinsamen ‚Component' hin verschoben.

- **Broken-Line**

  Nur in der Nähe eines vorherigen Spurrandpunktes wurde eine passende ‚Component' gefunden oder es wurden unterschiedliche ‚Components' für die zwei vorherigen Spurrandpunkte gefunden. Die Spurmarkierung scheint unterbrochen zu sein und die Spur nimmt den Zustand ‚Broken-Line' an. Es wird die flächenmäßig größere der beiden ‚Components' ausgewählt und der naheliegende Spurrandpunkt passend verschoben. Die neue Position des weiter entfernten Spurrandpunktes wird wieder über eine Funktion dritten Grades ermittelt, welche aus den Punkten der ausgewählten ‚Component' approximiert wurde.

- **Vanished-Line**

  ‚Vanished-Line' beschreibt den Zustand einer Spurmarkierung, wenn die Spurrandpunkte nah beieinanderliegen und in ihrer Nähe kein passendes Label gefunden wurde. In diesem Fall befindet sich das Fahrzeug an einer Kurve und die innenliegende Spurmarkierung verschwindet ‚geplant' vom Kamerabild. Die Spurrandpunkte verbleiben an ihren Positionen und die Spur wird markiert, damit die Lenkregelung am Fahrzeug weiß, dass die Spur verschwunden ist und nicht beachtet werden sollte.

- **Lost-Line**

  Die ‚Lost-Line' ist eine Abwandlung der ‚Vanished-Line', wobei die Spurrandpunkte nicht nah genug beieinanderliegen, um eine Kurvenfahrt abzubilden. Das Verhalten ähnelt dem der ‚Vanished-Line', abgesehen davon, dass der Suchraum um die Spurrandpunkte bei jedem neuen Videobild vergrößert wird, um die verlorene Spur schnellstmöglich wiederzufinden.

Grundsätzlich werden die rechten und linken Spurenmarkierungen separat erkannt und verfolgt. Beim Versetzen der Spurrandpunkte wird allerdings darauf geachtet, dass sich die rechts bzw. links zugeordneten Punkte nicht zu nahekommen oder gar überkreuzen.

In Abbildung 17 wird ein Testbild der Spurerkennung dargestellt. Zu sehen ist eine Rechtskurvenfahrt. Das IPM-Bild wurde binarisiert, weswegen nur schwarze oder weiße Bildpunkte zu sehen sind. Rot und grün markiert sind die Spurrandpunkte der rechten und linken Spurmarkierung. Mit Orange sind für beide Spurmarkierungen die approximierten Funktionen eingezeichnet, die bei der Bestimmung neuer Spurrandpunkte verwendet werden können.

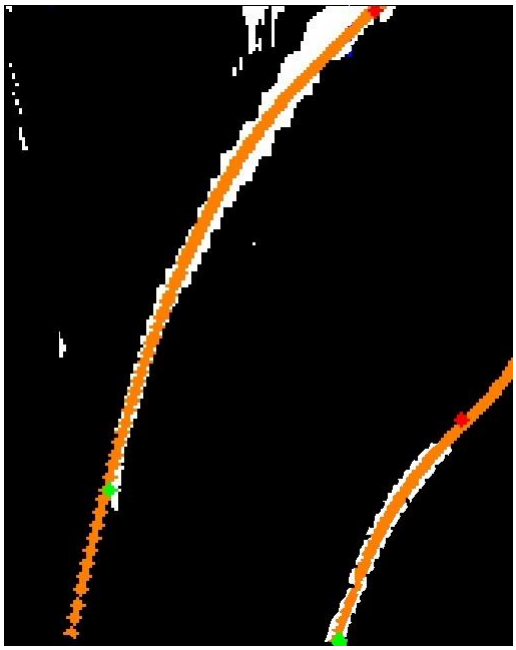

Abbildung 17 – Testbild der bisher verwendeten Spurerkennung.
Spurrandpunkte sind rot und grün markiert. Die approximierte Spurmarkierungsfunktion dritten Grades wird orange dargestellt.

4.1.2 Schwächen des bisher eingesetzten Spurerkennungsalgorithmus

Der soeben vorgestellte, bisher vom Team Smart-Driving genutzte Spurerkennungsalgorithmus hat sich im AADC 2015/16 als eine Schwachstelle der Fahrzeug-KI herausgestellt. Da die Spurerkennung, wie unter 4.1 erläutert, einen zentralen und wichtigen Bestandteil des autonomen Fahrens darstellt, trug die bisher verwendete Spurerkennung zu einem Großteil dazu bei, dass das Team Smart-Driving nicht mit den Top-Teams des AADC mithalten konnte.

Bei der Fehleranalyse nach dem AADC 2015/16 wurden folgende Schwachstellen des bisher genutzten Spurerkennungsalgorithmus festgestellt:

- **Erkennen von Spurmarkierungen**

  Nach dem Schritt des ‚Connected Component Labeling' wurden nur die ‚Connected Components' zur Spurerkennung herangezogen, welche bestimmte Eigenschaften erfüllt haben. Diese Eigenschaften wurden experimentell für verschiedene Szenen bestimmt, indem als falsch erkannte Spurmarkierungen nach Merkmalen untersucht wurden, die sie von echten Spurmarkierungen unterschieden. So wurde eine Liste an Eigenschaften zusammengestellt, die die ‚Connect Components' erfüllen mussten, um als Spurmarkierung erkannt zu werden.

  Leider erwiesen sich diese Eigenschaften sowohl als zu streng, als auch zu lasch, was zu Fehlerkennung führt. Die Liste der Eigenschaften konnte aber nicht erweitert oder gekürzt werden, ohne dabei bereits korrekt erkannte oder verworfene ‚Connected Components' wieder falsch zu klassifizieren. Dadurch unterliefen der Spurerkennung Fehler bei der Initialisierung und bei hellen Hindernissen.

- **Missinterpretation von Kreuzungen**

  Bei der Betrachtung von Kreuzungen und Kurven im IPM-Bild fällt auf, dass diese sich in ihren Spurmarkierungsverlauf ähneln. Beide Markierungen verlaufen grundsätzlich gebogen. Kreuzungen erscheinen durch ihre rechtwinkligen Geraden jedoch abrupter im IPM-Bild als Kurvenzüge.

  Die bisher genutzte Spurerkennung interpretierte Kreuzungen aufgrund ihrer Ähnlichkeit meist als Kurven. Dies resultierte im automatischen Abbiegevorgang des AADC-Fahrzeugs an Kreuzungen, was in den meisten Fällen ein Fahrfehler war.

- **Festlaufen in fehlerhaften Zuständen**

  Ein weiterer Schwachpunkt der bisher verwendeten Spurerkennung war, dass sich die Spurrandpunkte nur schwer aus fehlerhaften Zuständen befreien konnten. Damit ist gemeint, dass durch Fehlerkennungen Spurrandpunkte an falsche Randstellen verschoben wurden. Dort haben sie im Umkreis ihre eigentliche Spurmarkierung nicht schnell genug wiedergefunden, um das Fahrzeug am Verlassen der Fahrspur zu hindern.

Um diesem Zustand vorzubeugen, wurde die Spurerkennung regelmäßig zurückgesetzt. Dadurch traten allerdings Probleme bei der Initialisierung in Kurven oder an Kreuzungen auf.

- **Initialisierung nur auf Geraden möglich**

  Wie zu Beginn des Kapitels 4.1.1.3 erwähnt, initialisierte sich der Spurerkennungsalgorithmus, durch die Suche nach langen, geraden Spurmarkierungen, welche vollständig links bzw. rechts der Bildmitte liegen. Die bereits erwähnten Einschränkungen bei der Klassifikation von ‚Connected Components' sorgten dafür, dass sich der Algorithmus nur auf Geradenstücken sinnvoll initialisieren konnte.

  Dies war vor allem deswegen problematisch, weil die Spurerkennung zum Schutz vor dem Festlaufen in fehlerhaften Zuständen nach jeder Fahraufgabe (Abbiegen, Ausparken) zurückgesetzt wurde. Begegnete das AADC-Fahrzeug des Teams Smart-Driving also direkt nach einer solchen Fahraufgabe einer Kreuzung oder eine Kurve, musste das Fahrzeug reglos stehen bleiben. Es konnte keine neue Spur finden.

- **Irritation durch helle Hindernisse**

  Durch die zuvor erwähnten Probleme bei der Klassifizierung der ‚Connected Components' traten bei der bisher genutzen Spurerkennung immer dann Probleme auf, wenn helle Hindernisse am Fahrbahnrand auftauchten. Sahen diese durch die Verzerrung im IPM-Bild einer Fahrbahnmarkierung ähnlich, konnten die Spurrandpunkte am IPM-Rand verrutschen. Durch das weiter oben beschriebene Festlaufen bei Fehlern konnte sich die Spurerkennung aus solchen Situation nicht eigenständig befreien.

- **Spurverlust beim Verdecken der Spurmarkierungen**

  Hindernisse auf der Fahrbahn, welche die Spurmarkierungen teilweise oder vollständig verdeckten, stellten für die aktuelle Spurmarkierung eine unlösbare Aufgabe dar. Da die Hindernisse nicht plötzlich auftauchten, sondern durch die Fahrtbewegung ins Bild geschoben wurden, vollzog der Spurerkennungsalgorithmus die Verschiebung der Spurrandpunkte am IPM-Rand nach. Dadurch geriet die Spurerkennung wieder in einen fehlerhaften Zustand, wodurch das Fahrzeug von der Fahrbahn abkam.

Aufgrund der fünf dargestellten Schwachpunkte der bisher genutzten Spurerkennung wurde im Team Smart-Driving entschieden, dass zum AADC 2017 ein neuer Algorithmus zum Einsatz kommen muss. Dieser soll im Zuge dieser Arbeit entwickelt werden. Dazu werden im kommenden Abschnitt zunächst die Anforderungen an die neue Spurerkennung spezifiziert.

4.2 Anforderungen an die neue Spurerkennung

Bevor mit der Entwicklung eines neuen Spurerkennungsalgorithmus begonnen werden kann, müssen die Anforderungen an einen solchen festgelegt werden. Dabei kann auf der soeben vorgestellten Liste an Schwachstellen der alten Spurerkennung aufgebaut werden. Folgende Anforderungen wurden in Zusammenarbeit mit dem Team Smart-Driving als zwingend notwendig angesehen:

1. **Erkennen und Verfolgen von durchgezogenen Spurmarkierungen**

   Als Grundanforderung an eine Spurerkennung gilt, dass der Algorithmus die in einem vorgegebenen Videobilderstrom enthaltenen Spurmarkierungen erkennt und markiert. Im hier gegebenen, speziellen Fall soll das Video Streckenabschnitte aus dem AADC enthalten. Außerdem muss der Algorithmus vorerst nur die durchgezogenen, äußeren Spurmarkierungen erkennen.

2. **Stabilität bei Kreuzungen**

   Eine Schwäche der bisherigen Spurerkennung war, dass Kreuzungen fälschlicherweise als Kurven interpretiert wurden. In Bezug auf die bisherigen Kreuzungsmöglichkeiten beim AADC (Plus- und T-Kreuzung) sollen Kreuzungen generell als Geraden interpretiert werden. Der Abbiegevorgang ist explizit nicht Teil der Spurerkennung.

3. **Stabilität beim Verdecken von Spurmarkierungen**

   Sollten aufgrund von Hindernissen oder Fahrbahnabnutzung Spurmarkierungen verdeckt bzw. verschwunden sein, darf dies die Spurerkennung nicht irritieren oder gar zu falschen Angaben verleiten. Der bisherige Spurmarkierungsverlauf soll in einem solchen Fall beibehalten werden, bis die Markierungen wieder zu erkennen sind. Dies bedeutet, dass auf Geraden die verdeckte Spurmarkierung als gerade Markierung weitergeführt werden soll. In Kurven soll der Algorithmus den zuletzt erkannten Kurvenradius während der Verdeckung beibehalten.

4. **Leichte Austauschbarkeit des Algorithmus**

   Neben den funktionalen Anforderungen wurde vom Team Smart-Driving auch eine Anforderung bezüglich der Softwarearchitektur gestellt, in die die Spurerkennung eingebettet werden soll: Es soll eine Softwarefassade vor dem Algorithmus implementiert werden, die eine leichte Austauschbarkeit des Spurerkennungsansatzes ermöglicht.

Als optionale Anforderungen wurden festgelegt:

5. **Erkennen von unterbrochenen Spurmarkierungen**

   Als Soll-Anforderung an die Spurerkennung wurde bereits das Erkennen von durchgezogenen Spurmarkierungen definiert. Falls möglich, soll die neue Spurerkennung optional auch unterbrochene Linien erkennen können. Dazu zählt bspw. die mittlere Spurmarkierung.

6. **Unabhängigkeit von der Kamera**

   Wie bereits unter 3.3.3 beschrieben, existieren im Augenblick zwei denkbare Kameramodelle, die das Videobild als Grundlage für die Spurerkennung erzeugen können. Um einen schnellen Austausch der Kameras zu ermöglichen, wäre es nützlich, wenn der Algorithmus ohne Änderungen mit beiden Kamerasystemen funktionieren würde.

7. **Ausgabe bei unsicherem Verhalten**

   Durch Verdeckung von Fahrbahnmarkierungen kann der Fall eintreten, dass die Spurerkennung die Position der Straßenlinien nur noch schätzen kann. Diese unsicheren Ausgaben sollten entsprechend gekennzeichnet werden. Dadurch wird verhindert, dass Fehler der Spurerkennung zum Regelungssystem, das die Lenkung des Fahrzeugs steuert, weitergeleitet werden.

8. **Erkennen von Kreuzungen**

   Neben dem sicheren Überfahren von Kreuzungen als Soll-Anforderung, wäre ein Erkennen von Kreuzungsbereichen für das Team Smart-Driving nützlich. Im Augenblick werden Kreuzungen über Straßenschilder durch die KI erkannt. Die zusätzliche Erkennung über Fahrbahnmarkierungen kann die Robustheit der Kreuzungserkennung erhöhen.

9. **Erkennen von Parklücken**

   Ähnlich wie Kreuzungen werden Parklücken in der aktuellen Fahrzeug-KI nur über Schilder und Ultraschallsensoren erkannt. Eine Identifizierung von potenziellen Parklücken anhand von Fahrbahnmarkierungen über den Spurerkennungsalgorithmus könnte ein präziseres Anfahren der Parkplätze ermöglichen.

Auf Grundlage der soeben festgelegten Anforderungen kann im Folgenden mit der Planung der neuen Spurerkennung begonnen werden.

4.3 Planung

Der Kernaspekt bei der Planung der neuen Spurerkennung ist die Recherche nach einem geeigneten Algorithmenansatz. Dabei fließen als Kriterien zum einen die im Kapitel 4.2 festgelegten Anforderungen in die Auswahl ein. Zum anderen müssen die Algorithmen hinsichtlich ihrer Umsetzbarkeit auf dem Modellfahrzeug der HTWK geprüft werden. Setzt ein bestimmter Spurerkennungsansatz bspw. eine Hardware voraus, die nicht auf dem autonomen Auto verbaut ist, scheidet dieser aus der Planung aus.

Das Institute of Electrical and Electronics Engineers (IEEE) ist ein Berufsverband von Ingenieuren. Wissenschaftliche Beiträge, welche über Zeitschriften oder Konferenzen des IEEE publiziert wurden, gelten als besonders hochwertig. Eine Suchanfrage auf dem IEEE-Portal mit dem Begriff ‚Spurerkennung' liefert über 400 Artikel, welche sich in den letzten zehn Jahren mit diesem Thema beschäftigt haben. Daran ist zu erkennen, wie intensiv auf dem Gebiert der Spurerkennung und -verfolgung geforscht wird.

Aus diesem Grund entschied sich der Autor dieser Arbeit dafür, einen renommierten Algorithmenansatz aus dem Archiv des IEEE als Grundlage für die neue Spurerkennung des Teams Smart-Driving zu verwenden. In dieser Arbeit wird deshalb der Algorithmus von [JYS16] näher betrachtet. Kapitel 4.3.1 beleuchtet die genaue Funktionsweise dieses Spurerkennungsansatzes, während der darauffolgende Abschnitt 4.3.2 die Gründe für dessen Verwendung erläutert.

4.3.1 Funktionsweise der spatiotemporalen Spurerkennung

[JYS16] nutzen für ihre vorgestellte Spurerkennung sogenannte ‚spatiotemporale' Bilder. Das englische Adjektiv ‚spatiotemporal' bedeutet ins Deutsche übersetzt in etwa ‚raumzeitlich'. Diese Bedeutung lässt sich auch auf die von den Autoren vorgestellte Spurerkennung übertragen: Sie überwacht einen festgelegten Bildraum über einen gewissen Zeitabschnitt, um dadurch leichter Spurmarkierungen ausmachen zu können.

Im Detail betrachtet funktioniert dieser Ansatz wie folgt: Es wird versucht, auf einer beliebigen Bildzeile die Spurmarkierungen zu identifizieren und zu verfolgen. Dazu wird diese Bildzeile aus jedem Bild des Videobildstroms extrahiert und zeitlich geordnet in ein eigenes Bild eingefügt. Dadurch entsteht, bildlich gesprochen, ein Stapel von Bildzeilen, der ein neues Bild ergibt. Dieses neue Bild entspricht dem zeitlichen Verlauf der betrachteten Zeile im Videobild, weswegen das neue Bild als ‚spatiotemporal' (raumzeitlich) bezeichnet werden kann.

Kameras in autonomen Fahrzeugen sind grundsätzlich festmontiert und bewegen sich während der Fahrt kaum. Dadurch bleibt der Sichtbereich und der Blickwinkel der Kameras unverändert, was bei der bisher verwendeten Spurerkennung des Teams Smart-Driving, bspw. bei der Fluchtpunkterkennung, ausgenutzt wurde. Auch für die spatiotemporale Spurerkennung wird, ausgehend vom Fakt der unbeweglichen Kamera, von [JYS16] folgende Hypothese formuliert:

Ändert sich der Blickwinkel der Kamera wenig bis gar nicht, entspricht eine Zeile des Kamerabildes immer ungefähr dem gleichen Abstand zum Fahrzeug wie die Zeilen auf gleicher Höhe in vorherigen oder kommenden Kamerabildern. Auf Dauer sollten demnach alle Spurmarkierungen, die durch die betrachtete Zeile des Kamerabildes verlaufen, den gleichen Abstand haben. Diese zeitliche Konstanz nutzt der Spurerkennungsalgorithmus aus. Werden die Zeilen von aufeinanderfolgenden Kamerabildern nämlich übereinandergestapelt, so sollten alle Spurmarkierungen vertikal im aus dem Stapel resultierenden Bild verlaufen. Diese vertikalen Gerade sind leicht zu entdecken und zu verfolgen, was wiederum die Spurerkennung und -verfolgung im Kamerabild ermöglicht.

Abbildung 19 zeigt ein spatiotemporales Bild, welches auf der Bildzeile eines Videos von einer Teststrecke des Teams Smart-Driving beruht. Abbildung 18 ist das dazu passende Original-Kamerabild. Darin ist mit rot die Zeile markiert, auf der das spatiotemporale Bild beruht.

Abbildung 18 – Kamerabild einer Kurvenfahrt.
Die Bildzeile, die für die Beispiele des spatiotemporalen Algorithmus verwendet wurde, ist rot markiert.

Abbildung 19 – Unausgeglichenes, spatiotemporales Bild.
Basierend auf der markierten Bildzeile aus Abbildung 18.

Abbildung 19 zeigt aber auch, dass die oben aufgestellte Hypothese nur bedingt korrekt ist. Bei einer Autofahrt kann ein Fahrzeug aufgrund von Bodenbeschaffenheiten niemals komplett gerade gefahren werden. Leichte Ausgleichbewegungen nach links und rechts sind normal. Diese sollten also auch im spatiotemporalen Bild beachtet werden. Dazu raten [JYS16] aufeinanderfolgende Bildzeilen nicht direkt übereinandern zu stapeln, sondern sie dabei auch leicht gegeneinander zu verschieben.

Diese Verschiebungen sind außerdem nötig, um die Lenkbewegungen bei Kurvenfahrten auszugleichen. Nur mit passenden Verschiebungen bleiben die Spurmarkierungen innerhalb von Kurven im spatiotemporalen Bild vertikal.

Um die beste Verschiebung zu bestimmen, werden zunächst alle Verschiebungen in einem vorher festgelegten Rahmen ausprobiert. Als Gütekriterium der Verschiebung wird die absolute Differenz der Grauwerte der aktuellen und vorhergehenden Bildzeilen betrachtet. Umso kleiner diese Differenz ist, desto besser passen die Zeilen zusammen. Die Verschiebung mit der geringsten Differenz wird als beste Verschiebung akzeptiert.

Abbildung 20 zeigt das gleiche spatiotemporale Bild wie Abbildung 19, nur dass diesmal passende Verschiebungen durchgeführt wurden. Da sich das Fahrzeug auf der rechten Fahrspur befindet, sind die mittleren und rechten Spurmarkierungen die interessantesten Linien und sollten demnach möglichst vertikal verlaufen. Es ist erkennbar, dass ebendiese Fahrbahnmarkierungen vertikaler und mit weniger Schwingung verlaufen als im unausgeglichenen, spatiotemporalen Bild aus Abbildung 19.

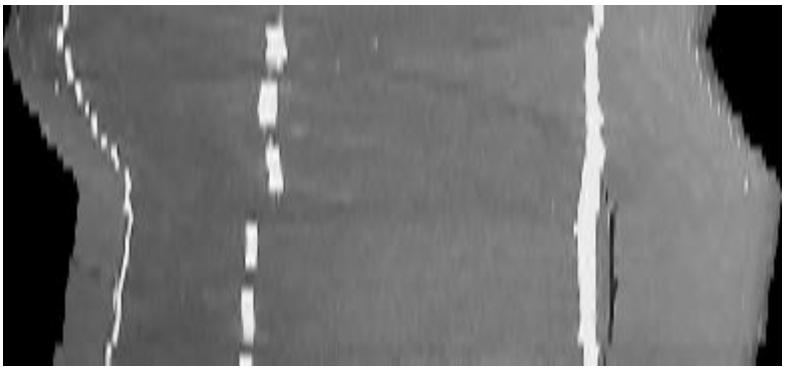

Abbildung 20 – Ausgeglichenes, spatiotemporales Bild.
Die interessanteren mittleren und rechten Spurmarkierungen verlaufen vertikaler als in Abbildung 19.

Ausgehend vom durch Verschiebungen ausgeglichenen, spatiotemporalen Bild kann nun die Position der Spurmarkierung auf der Bildzeile bestimmt werden. Dazu wird das spatiotemporale Bild, ähnlich wie das IPM-Bild der bisher genutzten Spurerkennung, binarisiert (s. Kapitel 2.2.2.3). Dadurch werden die Spurmarkierungen hervorgehoben. Mit dem Hough-Algorithmus (s. Kapitel 2.2.2.1) werden die vertikalen Linien des Binärbildes bestimmt. In der Nähe der Schnittpunkte der dominantesten Hough-Linien mit dem oberen Rand des spatiotemporalen Bild kann nun nach weißen Bereichen in der aktuellsten, binarisierten Bildzeile gesucht werden. Diese Bereiche sind die Spurmarkierungen. Ihre Positionen können unter Abzug der Ausgleichsverschiebung wieder auf das Originalbild übertragen werden.

Abbildung 21 zeigt ein binarisiertes, spatiotemporales Bild. Darin sind die vertikalen Hough-Linien blau und die stärksten vertikalen Hough-Linien grün markiert. Die gefundenen Spurmarkierungen werden durch rote Punkte am oberen Bildrand symbolisiert.

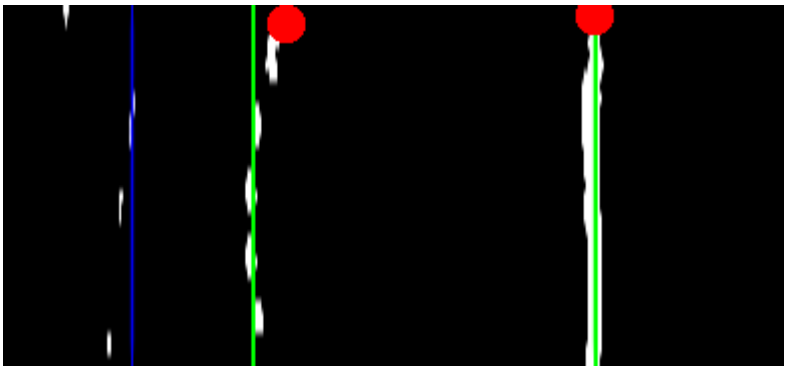

Abbildung 21 – Binarisiertes, spatiotemporales Bild (vgl. Abbildung 20).
Vertikale Hough-Linien sind blau bzw. grün markiert. Die erkannten Spurpunkte entsprechen den roten Punkten.

Sollte der Algorithmus bei der Suche nach den Spurmarkierungen keine weißen Bereiche in der aktuellsten, binarisierten Bildzeile finden, stellt dies kein Problem dar. Aufgrund der zeitlichen Konstanz des spatiotemporalen Bildes kann der Algorithmus den Schnittpunkt der dominantesten Hough-Linien mit dem oberen Rand des spatiotemporalen Bildes als geschätzten Punkt der Spurmarkierung angeben. Dieser Punkt sollte nah an der realen Spurmarkierung liegen. Algorithmen, die auf den Spurerkennungsergebnissen aufbauen, können die Information, ob ein Spurmarkierungspunkt geschätzt wurde, für sich nutzen. Der Steuerungsalgorithmus, der die Lenkung des Fahrzeugs bestimmt, kann geschätzte Punkte bspw. weniger stark gewichten.

Der vorgestellte Spurerkennungsansatz wird nun gleichzeitig für mehrere Bildzeilen eines Videobildes durchgeführt werden. Dabei werden alle spatiotemporalen Bilder prinzipiell unabhängig voneinander berechnet. Aus jedem spatiotemporalen Bild kann ein linker und rechter Spurmarkierungspunkt abgelesen werden. Mithilfe dieser Menge an Spurmarkierungspunkten können bspw. mathematische Funktionen angenähert werden, die sowohl gerade, als auch gekrümmte Spurmarkierungen abbilden können. Die Nutzung mehrerer spatiotemporaler Bilder ermöglicht dem Algorithmus also neben der Erkennung von Geraden auf die Erkennung von Kurven.

Abbildung 22 zeigt hierzu erneut das Videobild aus Abbildung 18, dabei aber mit den erkannten Spurmarkierungen aus fünf Bildzeilen bzw. spatiotemporalen Bildern. Rote Punkte sind sicher erkannte Spurmarkierungen. Bei grünen Punkten musste die Spurerkennung die Position der Markierungen schätzen. Das Bild zeigt eine Kurve, in der der Algorithmus die gekrümmte Fahrbahnmarkierung durch die fünf einzelnen Punkte annähern kann.

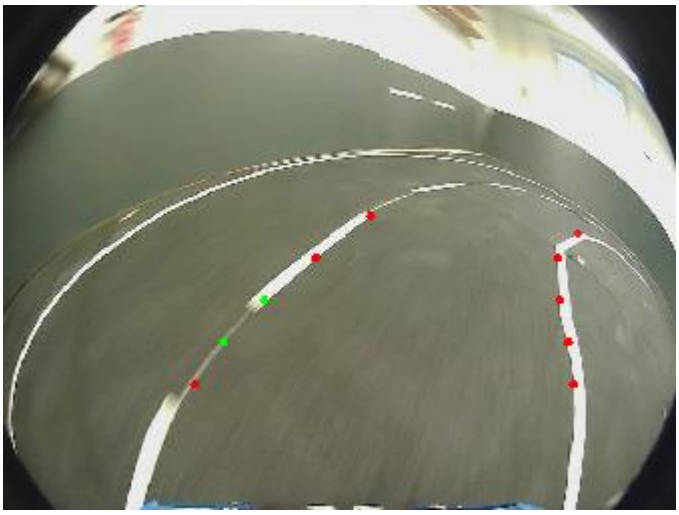

Abbildung 22 – Kamerabild aus Abbildung 18 mit detektierter Spurmarkierung.
Es wurden sechs spatiotemporale Bilder eingesetzt. Rote Punkte entsprechen sicheren Spurmarkierungen, grüne Punkte wurden geschätzt.

4.3.2 Gründe für die Verwendung der Spatiotemporal-Spurerkennung

Im vorangegangenen Kapitel wurde der einzigartige, raumzeitliche Ansatz des Algorithmus von [JYS16] erläutert. Da er verspricht, fast alle notwendigen und optionalen Anforderungen des Teams Smart-Driving zu erfüllen, bietet er sich als Grundlage der neuen Spurerkennung an. Die Gründe für seine Verwendung werden im Folgenden nochmals genauer erläutert:

- **Erkennen und Verfolgen von Spurmarkierungen**

  Der spatiotemporale Spurerkennungsansatz verspricht neben dem Erkennen von durchgezogenen Linien auch das Erkennen der gestrichelten Spurmittellinien (vgl. [JYS16] Abb. 5) . Dadurch kann eine exaktere Positionierung des Fahrzeugs auf der Fahrspur erfolgen, da Fehlabschätzungen über die äußeren Spurmarkierungen vermieden werden.

- **Stabilität bei unterbrochenen oder verdeckten Spurmarkierungen**

  Durch den Einfluss des zeitlichen Verlaufs der Spurmarkierungen gilt die spatiotemporale Spurerkennung als besonders robust gegenüber Spurunterbrechungen durch Kreuzungen oder Hindernisse (vgl. [JYS16] Kap. II.E.4, Kap. III). Damit verspricht sie einen Vorteil gegenüber der bisher verwendeten Spurerkennung.

  Robust bedeutet in diesem Fall, dass der Algorithmus die Spurpunkte verdeckter Markierungen über einen festgelegten Zeitraum (meist zwei Sekunden, s.a. Kapitel 4.5.3) abschätzen kann. Die geschätzten Punkte sind dabei so genau, dass das Fahrzeug innerhalb des festgelegten Zeitraums nicht von der korrekten Fahrspur abkommt.

- **Unabhängigkeit vom Kamerasystem**

  Da der spatiotemporale Algorithmus nur mit einzelnen Bildzeilen des originalen Kamerabildes arbeitet, ist er völlig unabhängig vom Kamerasystem. Aus dem gleichen Grund ist er robust gegen Verzerrungslinsen wie die der Fish-Eye-Kamera. Dadurch ist er für die Forschung des Teams HTWK Smart-Driving mit den verschiedenen Kamerasystemen geeignet.

- **Ausgabe bei unsicherem Verhalten**

  Der Algorithmus erkennt, wenn er einen Spurmarkierungspunkt nur anhand der Hough-Line bestimmt hat, ohne dabei auf eine Spurmarkierung im aktuellen Bild zu treffen. Dieser Zustand, von [JYS16] als ‚geschätzt' (engl. ‚guessed') bezeichnet, tritt genau dann ein, wenn der Algorithmus sich selbst unsicher ist, ob er die Spurmarkierung korrekt bestimmt hat. Bei längerem unsicheren Verhalten könnten die Spurhaltesysteme das autonome Fahrzeug stoppen und die Spurerkennung zurücksetzen, um wieder korrekte Aussagen über die Spurmarkierungen zu erhalten.

Aufgrund der soeben erläuterten Gründe wurde für diese Arbeit entschieden, die spatiotemporale Spurerkennung für das Team Smart-Driving umzusetzen. Die Übertragung des vorgestellten, theoretischen Ansatzes in den Quellcode, welcher auf dem im Zuge dieser Arbeit konstruierten Modelfahrzeug funktioniert, wird im folgenden Kapitel erläutert.

4.4 Umsetzung

Soeben wurde die Planung der neuen Spurerkennung mit der Entscheidung abgeschlossen, dass der spatiotemporale Ansatz als Grundlage der neuen Spurerkennung dienen soll. Dieser theoretische Ansatz muss nun für das Softwaresystem des Teams Smart-Driving umgesetzt werden, um auf dem neu konstruierten Modellfahrzeug funktionieren zu können. Dies wird in den folgenden Abschnitten beschrieben.

4.4.1 Grundarchitektur

Wie bereits unter Kapitel 3.3.5 erwähnt, baut das autonome Fahrsystem des Teams Smart-Driving auf einer bestehenden Middleware auf. Diese Middleware kann einerseits das im AADC verwendete und von Audi präferierte ADTF sein. Andererseits steht mit ROS eine Open-Source Alternative zur Verfügung, welche auf dem neu konstruierten Modellfahrzeug Verwendung findet. Beide Middlewares verbindet, dass sie Kommunikationsmöglichkeiten zwischen unabhängigen Softwaremodulen zur Verfügung stellen. Diese können die Entwickler, wie bspw. das Team Smart-Driving, nutzen, um einen Datenaustausch zwischen den verschiedenen Bestandteilen des autonomen Fahrsystems umzusetzen. Außerdem bieten sie Gerätetreiber für die in den Modellfahrzeugen verwendete Hardware an.

Für die Spurerkennung ist die interessante Hardware die Kamera, welche einen Videobilderstrom der Straße vor dem Fahrzeug liefert. ROS und ADTF nutzen zum Abgreifen und Bereitstellen des Videobildes jeweils eigene Datenformate. Das Spurerkennungssystem sollte also softwareseitig eine von der Middleware unabhängige Schnittstelle zur Verfügung stellen. Dadurch kann sie sowohl im ADTF, als auch im ROS verwendet werden, ohne intern verändert werden zu müssen.

Um diese universelle Schnittstelle umzusetzen, wurde für die Entwicklung des neuen Spurerkennungsalgorithmus die ‚Open Computer Vision'-Bibliothek (OpenCV) verwendet. OpenCV ist eine in C++ geschriebene Softwarebibliothek für digitale Bildverarbeitung und die eng verwandte Computer Vision. Sie beinhaltet einerseits Definitionen für Datenformate, welche die Infrastruktur der Bibliothek bereitstellen. Dazu zählen bspw. Matrizen, welche auch als Bilddatenformat genutzt werden, Vektoren und geometrische Figuren. Andererseits stellt OpenCV vorimplementierte Verfahren der digitalen Bildverarbeitung bereit. In OpenCV ist bspw. der Hough-Algorithmus bereits enthalten.

Dabei sind alle Algorithmen in OpenCV darauf ausgelegt, möglichst effizient und ressourcensparend berechnet werden zu können (vgl. [BK08] S. 1 ff.). Seit Version 2.4 ist es sogar möglich, OpenCL in OpenCV zu nutzen. OpenCL ist ein offener Quellcodestandard, der es ermöglicht, Code zu schreiben, der die gesamte Rechenleistung von heterogenen Systemen nutzen kann. Im Falle von OpenCV können so Algorithmen der digitalen Bildverarbeitung auf dem Grafikprozessor berechnet werden. Der Grafikprozessor kann diese Berechnungen effizienter durchführen und gleichzeitig dem normalen Prozessor Rechenlast abnehmen (vgl. [Op16]).

Deshalb hat die Verwendung von OpenCV als Grundlage für den Spurerkennungsalgorithmus gleich zwei Vorteile: Zum einen wird über die vorgegebenen Datenstrukturen der Bibliothek eine einheitliche Schnittstelle für die beiden Middleware-Systeme ROS und ADTF bereitgestellt. Zur Verwendung der Spurerkennung müssen dann nur die Bildformate der Middlewares in die OpenCV-Bildformate transformiert werden. Zum anderen werden durch die spezielle Programmierung der Algorithmen von OpenCV und die Nutzung von OpenCL die knappen Ressourcen des Bordcomputers des autonomen Modellfahrzeugs geschont.

Eine Anforderung an die neue Spurerkennung war, dass die Implementation des eigentlichen Algorithmus hinter einer austauschbaren Fassade zu verbergen ist (vgl. 4.2 Anforderung 4). Mithilfe von OpenCV ist dies nun möglich: Ein Spurerkennungsalgorithmus erhält als Eingabe ein OpenCV-Bild. Die Ausgabe des Algorithmus bestehe aus zwei Listen von Spurpunkten. Eine Liste repräsentiert dabei die linke, die andere Liste die rechte Spurmarkierung. Spurpunkte wiederum bestehen aus der Bildkoordinate, die der Algorithmus einer Spurmarkierung zugeordnet hat. Daneben enthalten sie eine Kennzeichnung, die angibt, ob der Punkt sicher gefunden oder geschätzt wurde. Die C++-Klassen und das zugehörige C++-Interface zeigt Quelltext 1.

```
#include <opencv2/opencv.hpp>

class LanePoint
{
 public:
  Point2f Point;
  bool Guessed;
}
class LanePoints
{
 public:
  vector<LanePoint> LeftPoints;
  vector<LanePoint> RightPoints;
}
class LaneDetector
{
 public:
  virtual LanePoints DetectLanes(const cv::Mat &input) = 0;
};
```

Quelltext 1 – Interfacedefinition der Spurerkennung

Zur besseren Veranschaulichung der Grundarchitektur dient außerdem Abbildung 23. Sie zeigt das vollständige Zusammenspiel aller Komponenten und Schnittstellen, die zur Ausführung der Spurerkennung notwendig sind. Der ROS-Master ist dabei der Kommunikationsserver, der alle ROS-Packages verbindet. Zu diesen Packages gehören der Kameratreiber ‚libuvc_ros‘, der die Bilddaten der Kameras ausliest, die Spurerkennung ‚lane_detection‘ und der darauf aufbauende, noch nicht existierende ‚lane_driver‘, der die Spurerkennungsdaten in Hardwareansteuerungen übersetzen soll. Im AADC würde der ROS-Master durch ADTF ersetzt werden.

Die Spurerkennung übersetzt die ROS-Bilddaten in OpenCV-Bilder und nutzt über die in Quelltext 1 definierte Schnittstelle den ‚SpatiotemporalLaneDetector‘. Dieser verwendet wiederum mehrere ‚SpatiotemporalImages‘, wie in Kapitel 4.3.1 erläutert. Die Ergebnisse der Spurerkennung werden als LanePoints an den ROS-Master geschickt, der sie an den ‚lane_driver‘ weiterleitet.

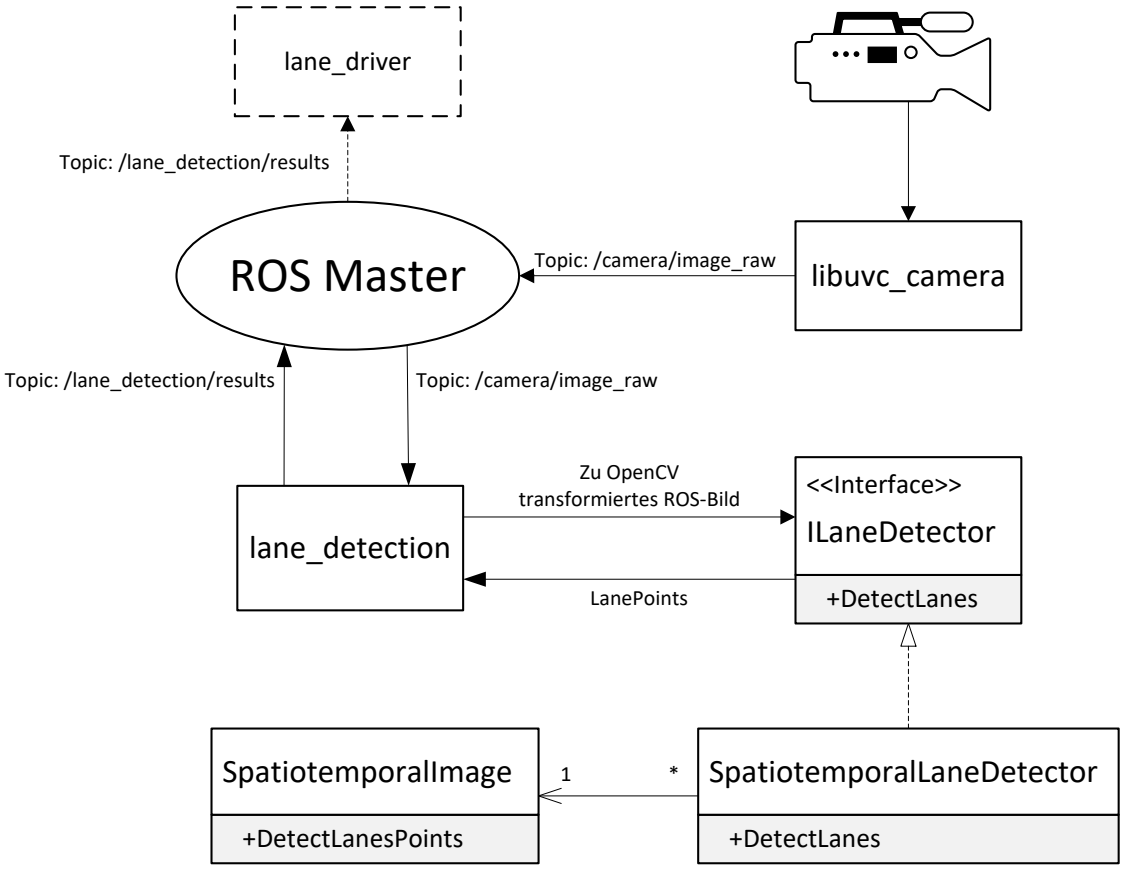

Abbildung 23 – Grundarchitektur der Spurerkennung

4.4.2 Spurerkennung laut Theorie

Die Umsetzung der Spurerkennung von [JYS16] erfolgte wie in Kapitel 4.3.1 beschrieben. Dabei musste beachtet werden, dass die Spurerkennung selbst aus mehreren spatiotemporalen Bildern besteht, die jedes für sich eine Bildzeile bearbeiten. Diese können aber nur gemeinsam eine Spurmarkierung annähern. Die Spurerkennung leitet demzufolge jedes Videobild an die spatiotemporalen Bilder weiter, die das Ergebnis des Algorithmus von [JYS16] auf ihrer Bildzeile berechnen. Alle Ergebnisse der spatiotemporalen Bilder werden dann von der Spurerkennung gesammelt und als Gesamtergebnis zurückgegeben. Um dies zu verdeutlichen, zeigt Quelltext 2 die Implementation des Spurerkennungsinterfaces für die spatiotemporale Bilderkennung. In diesem Fall werden fünf spatiotemporale Bilder verwendet.

```
LanePoints SpatioTemporalLaneDetector::DetectLanes(const Mat &input)
{
  LanePoints result;

  // Durchlaufe alle spatiotemporalen Bilder
  for (int imageIndex = 0; imageIndex < _spatioTemporalImages.size();
       ++imageIndex)
  {
    // Bestimme Spurmarkierung in spatiotemporalen Bild
    LanePointPair points =
        _spatioTemporalImages[imageIndex].DetectLanesPoints(input);
    result.LeftPoints.push_back(points.Left);
    result.RightPoints.push_back(points.Right);
  }
  return result;
}

SpatioTemporalLaneDetector::SpatioTemporalLaneDetector(int frameCount)
{
  // Registriere spatiotemporale Bilder mit Zeilenhöhe und Zeitspanne
  _spatioTemporalImages.push_back(SpatioTemporalImage(200, frameCount));
  _spatioTemporalImages.push_back(SpatioTemporalImage(240, frameCount));
  _spatioTemporalImages.push_back(SpatioTemporalImage(280, frameCount));
  _spatioTemporalImages.push_back(SpatioTemporalImage(320, frameCount));
  _spatioTemporalImages.push_back(SpatioTemporalImage(360, frameCount));
}
```

Quelltext 2 – Spatiotemporale Spurerkennungsimplementierung

Der eigentliche Algorithmus von [JYS16] wird im spatiotemporalen Bild abgearbeitet. Jedes spatiotemporale Bild kennt seine ihm zugewiesene Zeile Z des Originalbildes sowie die Anzahl an Zeilen n_{Bz}, aus denen es sich höchstens zusammensetzen soll. Mit diesen Parametern werden nun für jedes neue Videobild in allen spatiotemporalen Bildern die folgenden drei Phasen durchlaufen.

4.4.2.1 Einfügen der Bildzeile ins spatiotemporale Bild

Soll ein neues Videobild mithilfe eines spatiotemporalen Bildes nach Fahrspuren durchsucht werden, gilt es zunächst, die Bildzeile aus dem Videobild in das spatiotemporale Bild einzufügen.

Ein Videobild Q_t zum Zeitpunkt t sei ein Bild wie in Kapitel 2.2.2 definiert. Es habe demnach die Höhe h_Q und die Breite b_Q.

Sei $Bz_{Z,t}$ die Bildzeile des Videobildes Q_t auf der Höhe $Z \in \{0, \dots, h_Q - 1\}$ zum Zeitpunkt t. $Bz_{Z,t}(x)$ bezeichne demnach den Grauwert des Videobildes $Q_t(x, Z)$ zum Zeitpunkt t in Spalte x und Zeile Z.

Die Funktion $V(Z, t)$ bestimme die relative Verschiebung der Bildzeile zum Zeitpunkt t zu der Bildzeile zum Zeitpunkt $t - 1$. Die Verschiebung betrage dabei höchstens v_{max}.

$$V(Z,t) = \underset{v \in \{-v_{max}, \dots, v_{max}\}}{\operatorname{argmin}} \sum_{x=0}^{v+b_Q-1} \left| Bz_{Z,t}(x+v) - Bz_{Z,t-1}(x) \right|$$

Dabei gebe $Bz_{Z,t}(x)$ den Wert 0 zurück, wenn $x \notin \{0, \dots, b_Q - 1\}$.

Mithilfe der Verschiebung kann nun der Einfügepunkt $E_t \in \mathbb{Z}$ der aktuellen Bildzeile $Bz_{Z,t}$ in das bestehende spatiotemporale Bild bestimmt werden. Der Einfügepunkt zum Zeitpunkt t baut dabei auf dem Einfügepunkt zum Zeitpunkt $t - 1$ auf. Er verhält sich also rekursiv und sei wie folgt definiert:

$$E_t = \begin{cases} 0, f\ddot{u}r\ t \leq 0 \\ E_{t-1} + V(Z,t), sonst \end{cases}$$

Die Höhe des spatiotemporalen Bildes h_S entspricht der Anzahl an Bildzeilen n_{Bz}, aus denen es sich zusammensetzen soll. Dieser Parameter wurde bei der Erstellung des spatiotemporalen Bildes festgelegt. Die Breite des spatiotemporalen Bildes b_S bestimmt sich dynamisch anhand der extremsten Einfügepunkte für die letzten n_{Bz} Bildzeilen.

$$h_S = \min(n_{Bz}, t)$$

$$b_S = \left| \min_{t_{min} \in \{t, \dots, t - n_{Bz}\}} E_{t_{min}} - \max_{t_{max} \in \{t, \dots, t - n_{Bz}\}} \left(E_{t_{max}} + b_Q \right) \right|$$

Das spatiotemporale Bild S_t zum Zeitpunkt t mit Breite b_S und Höhe h_S sei demnach wie folgt definiert:

$$S_t(x,y) = \begin{cases} Bz_{Z,t-y}\left(x - E_{t-y}\right), f\ddot{u}r\ 0 \leq y \leq h_S \\ 0, sonst \end{cases}$$

Da OpenCV nicht mit negativen Einfügepunkten E_t umgehen kann, mussten diese im Quelltext korrigiert werden. Dazu wurden sämtliche Einfügepunkte der letzten n_{Bz} Bildzeilen um $|E_t|$ erhöht, falls $E_t < 0$.

4.4.2.2 Bestimmung der dominanten Linien

Bevor mit der Bestimmung der dominanten Linien im spatiotemporalen Bild begonnen werden kann, muss dieses zunächst vorverarbeitet werden. [JYS16] schlagen dazu einen Gauß-Filteroperation mit einer 11×11-Pixel großen Maske vor. Dieser Filter ist in OpenCV implementiert und konnte exakt so angewandt werden. Des Weiteren soll das spatiotemporale Bild mit einem adaptiven Grenzwert binarisiert werden. Da [JYS16] keine Methode zur Bestimmung des Grenzwertes angeben, wurde vom Autor dieser Arbeit der Yen-Algorithmus verwendet (vgl. [YCC95]).

Die eigentliche Bestimmung der dominanten Linien wurde mithilfe des Hough-Algorithmus durchgeführt, wie er in Kapitel 2.2.2.1 vorgestellt wurde. Zusätzlich wurde die Einschränkung vorgegeben, dass die gefundenen Linien nur einen Winkel $80° \leq \phi_V \leq 100°$ haben dürfen. Dadurch werden vom Hough-Algorithmus nur nahezu vertikale Linien gefunden. Die zwei stärksten Hough-Linien fließen als Parameter in die letzte Algorithmenphase ein.

4.4.2.3 Suchen von Spurmarkierungen

Die Suche nach Spurmarkierungen in der Bildzeile $Bz_{Z,t}$ zum aktuellen Zeitpunkt t geschieht folgendermaßen: Zunächst werden, wie im vorhergehenden Abschnitt beschrieben, die zwei dominantesten Hough-Linien HL_{t_1} und HL_{t_2} aus dem spatiotemporalen Bild S_t bestimmt. Da der Ablauf für beide Hough-Linien gleich ist, wird ab sofort allgemein HL_t betrachtet. Nun wird der Schnittpunkt dieser Linie mit der aktuellen Bildzeile berechnet. Da $Bz_{Z,t}$ in S_t die Zeile mit $y = 0$ ausmacht, ergibt sich die x_t^s-Koordinate des Schnittpunktes durch Umstellung der Hesse-Normalform der Linie (vgl. Kapitel 2.2.2.1) folgendermaßen:

$$d_t = x * \cos\phi_t + y * \sin\phi_t$$

$$x_t^s = \frac{d_t}{\cos\phi_t}$$

Die Bildzeile $Bz_{Z,t}$ wird anschließend pixelweise nach zusammenhängenden, weißen Abschnitten durchsucht. Die x-Koordinaten der Mittelpunkte dieser Abschnitte seien in der Menge $W \subset \mathbb{Z}$ enthalten.

Das Ergebnis des spatiotemporalen Spurerkennungsalgorithmus für die Zeile Z des Videobildes zum Zeitpunkt t ist die x-Koordinate der erkannten Spurmarkierung x_t^{Spur}:

$$x_t^{Spur} = E_t + \begin{cases} x_t^s, f\text{ür } W = \emptyset \\ \underset{x^w \in W}{\text{argmin}} |x_t^s - x^w|, sonst \end{cases}$$

x_t^{Spur} entspricht demnach jeweils dem Mittelpunkt eines weißen Abschnittes, der dem Schnittpunkt der obersten Bildzeile mit der dominanten Hough-Linie am nächsten liegt. Sollten keine weißen Abschnitte gefunden worden sein, ist x_t^{Spur} gleich x_t^s.

Beide Ergebnisse müssen abschließend mit dem Einfügepunkt E_t korrigiert werden, um auf das Videobild Q_t übertragen werden zu können.

Die soeben dargestellten Abläufe entsprechen den Ideen, die [JYS16] dargestellt haben. Bei ersten Tests des Algorithmus fielen allerdings Schwachstellen in eben diesem auf, die durch Anpassungen des Autors behoben wurden. Diese Schwachstellen und die entsprechenden Anpassungen der Spurerkennung werden im folgenden Kapitel vorgestellt.

4.4.3 Anpassungen

Während der Implementierung der Spurerkennung von [JYS16] musste regelmäßig getestet werden, ob der Algorithmus fehlerfrei funktioniert. Für erste Tests wurden dazu zwei Videos mit der Fish-Eye-Kamera aufgenommen, in denen das selbstkonstruierte Modellfahrzeug mit der Fernbedienung über einen Kurs gesteuert wurde (vgl. dazu Anhang 1 und Anhang 2). Die neue Spurerkennung wurde nun auf beide Videos angewendet, um die darin enthaltenen Spurmarkierungen anzuzeigen. Dabei zetraten Probleme beim Algorithmus von [JYS16] auf, nachdem dieser vollständig implementiert worden war. Die Entstehung und Behebung dieser Probleme wird im Folgenden erläutert.

4.4.3.1 Verwechslung der mittleren und äußeren Spurmarkierung

Das erste Problem, das bei der Betrachtung der Spurerkennung auf den Testvideos auffiel, war, dass die linken, äußeren Straßenmarkierungen als Mittellinie interpretiert wurden. Diese falsche Interpretation konnte nicht hingenommen werden, da sie das Spurhaltesystem dazu veranlassen könnte, die Fahrbahn zu verlassen.

Das Problem wurde dadurch behoben, dass statt der zwei dominantesten Hough-Linien HL_{t_1} und HL_{t_2} aus dem spatiotemporalen Bild S_t, zwei andere Hough-Linien zur Bestimmung von x_t^s verwendet wurden. Es wurden die zwei Hough-Linien ausgewählt, deren Abstand zur Bildmitte am geringsten ist. Diese Auswahl basiert auf der Annahme, dass die zwei Linien, die sich am nächsten an der Bild- und damit an der Fahrzeugmitte befinden, die Straßenmarkierungen sein müssen, die die Spur begrenzen.

Sei M_{HL} die Menge aller erkannten Hough-Linien im spatiotemporalen Bild S_t, welche durch das geordnete Paar (d, ϕ) definiert sind.

$$HL_t = \underset{(d_i,\phi_i)\in M_{HL}}{\operatorname{argmin}} \left| \frac{d_i}{\cos \phi_i} - \left(\frac{b_S}{2} + E_t \right) \right|$$

Mit der Verwendung dieser Hough-Linien konnten dauerhafte Fehlerkennungen korrigiert werden. Trotzdem traten an den planmäßigen Unterbrechungen der gestreiften Mittellinien weiterhin oben erwähnte Fehlinterpretationen auf. Diese wurden dadurch behoben, dass ein Maximalabstand x_{max}^w von x_t^s zu x^w eingeführt wurde. Sollte dieser Abstand überschritten werden, so wird der vorhergehende Spurpunkt x_{t-1}^{Spur} als Ergebnis verwendet. Dadurch werden Spurpunktsprünge verhindert. Es ergibt sich:

$$W_{max} = \{x^w \in W \mid x_{max}^w > |x_t^s - x^w|\}$$

$$x_t^{Spur} = E_t + \begin{cases} x_{t-1}^{Spur}, \text{für } W_{max} = \emptyset \\ \underset{x^w \in W_{max}}{\operatorname{argmin}} |x_t^s - x^w|, sonst \end{cases}$$

Mithilfe der vorgestellten Anpassungen konnte die Verwechslung von mittleren und äußeren Straßenmarkierung verhindert werden.

4.4.3.2 Verlust der Spurmarkierung zu Kurvenbeginn bzw. -ende

Ein weiterer, nachteiliger Effekt der unveränderten, spatiotemporalen Spurerkennung war, dass bei Fahrzeugeintritt in bzw. -austritt aus Kurven die Spurmarkierungen verloren wurden. Dies konnte darauf zurückgeführt werden, dass die Kurvenbewegung des Fahrzeugs zu einer plötzlichen Veränderung des spatiotemporalen Bildes führte. Diese Veränderung bewirkte, dass nicht alle Bereiche der Bildzeile stetig fortgeführt werden konnten. Die Bestimmung der Verschiebung der Scan-Line mittels minimaler Differenz ergab ein ungenügendes Ergebnis: Große Teile der Scan-Line passten im spatiotemporalen Bild zwar zur vorhergehenden Scan-Line, aber die wichtigen Bereiche, die Spurmarkierungen enthielten, wiesen Bildsprünge auf. Durch diese Sprünge war der Schnittpunkt der temporalen Hough-Line zu weit weg von der Spurmarkierung. Die Anpassung zur Verhinderung von Punktsprüngen sorgte im Anschluss dafür, dass die Spur nicht erkannt wurde.

Um dies zu verhindern, musste die Funktion $V(Z,t)$ so angepasst werden, dass sie Bereiche mit Spurmarkierungen bevorzugt verwendet, um die Verschiebung der Bildzeilen zu bestimmen. Durch den temporalen Aspekt der Spurerkennung lässt sich dazu x_{t-1}^{Spur} verwenden. Statt die gesamte Scan-Line zu vergleichen, wird nur ein kleiner Bereich der Größe $2 * x_{max}^{v}$ um x_{t-1}^{Spur} zur Bestimmung der Verschiebung benutzt.

$$V(Z,t) = \underset{v\in\{-v_{max},\dots,v_{max}\}}{\operatorname{argmin}} \sum_{x=\mathrm{v}+x_{t-1}^{Spur}-x_{max}^{v}}^{v+x_{t-1}^{Spur}+x_{max}^{v}} \left|Bz_{Z,t}(x+v) - Bz_{Z,t-1}(x)\right|$$

Damit konnte erreicht werden, dass bei Kurven die wichtigen Spurmarkierungen im spatiotemporalen Bild weiterhin gerade dargestellt und demnach erkannt werden.

Nachdem die Spurerkennung mit diesen Anpassungen auf den ersten beiden Testvideos zufriedenstellend funktionierte, konnte ein Praxistest der neuen Spurerkennung erfolgen. Wie dieser aussah und welche Erkenntnisse sich aus ihm ableiten ließen, beschreibt das nachfolgende Kapitel.

4.5 Praxistest der neuen Spurerkennung

Um beurteilen zu können, ob die neue Spurerkennung die an sie gestellten Anforderungen (vgl. Kapitel 4.2) erfüllen kann, musste ein Praxistest des Algorithmus erfolgen. Dazu wurden zunächst Videos des Kamerabildes des autonomen Modellfahrzeugs aufgenommen, während es mit der Fernbedienung über die Strecke gefahren wurde. Diese Videos stellten neben Rundfahrten auf einem Parcours auch Kreuzungssituationen oder das Umfahren von Hindernissen dar (vgl. Anhang 1 bis Anhang 6). Abschließend wurde der Algorithmus direkt auf dem autonomen Fahrzeug berechnet, während es via Fernbedienung über den Parcours gefahren wurde.

Bevor in diesem Kapitel geklärt wird, welche Anforderungen die neue Spurerkennung tatsächlich erfüllt und wo ihre Grenzen liegen, soll zunächst dargestellt werden, ob sich die Fish-Eye oder die Weitwinkelkamera besser für den angepassten Algorithmus von [JYS16] eignen.

4.5.1 Vergleich der Kamerasysteme

Wie bereits unter Kapitel 3.3.3 ausgeführt, wurden für das neue Modellfahrzeug des Teams Smart-Driving zwei neue Kamerasysteme vorgesehen: eine Fish-Eye- und eine Weitwinkel-Kamera. Anhand des neuen Spurerkennungsalgorithmus soll nun entschieden werden, welche der Kameras sich besser für das Fahrzeug eignet.

Rein vom theoretischen Blickwinkel betrachtet hat die Fish-Eye-Kamera den Vorteil des deutlich erhöhten Sichtwinkels im Vergleich zur Weitwinkelkamera. Mit diesem geht ein durch den Fish-Eye-Effekt verzerrtes Bild einher, das gerade Linien am Bildrand gekrümmt abbildet. Von der technischen Ausstattung weisen die beiden Kameras hinsichtlich der Bildauflösung und der Bildrate keine Unterschiede auf.

Weiterhin kann angeführt werden, dass die spatiotemporale Spurerkennung immun gegen die Verzerrung der Fish-Eye-Linse ist. Sie betrachtet nur einzelne Bildzeilen unabhängig voneinander und kann dadurch auch ohne Anpassungen auf Videobildern der Fish-Eye-Kamera arbeiten. Dadurch wird der Nachteil dieser Kamera im Vergleich zur Weitwinkel-Kamera aufgehoben.

Zusätzlich kann der Nutzen des erhöhten Sichtwinkels als Argument für die Fish-Eye-Kamera angebracht und durch folgende drei Gründe verdeutlicht werden:

- Durch das größere Sichtfeld kann die Straße direkt vor dem Fahrzeug betrachtet werden. Die Weitwinkelkamera zeigt die Straße erst ab ca. einem Meter vor dem Fahrzeug.
- In Kurvenfahrten verschwinden im Bild der Weitwinkelkamera ab einem bestimmten Punkt Teile der äußeren Spurmarkierungen vom Kamerabild. Dies hängt mit der Stellung des Fahrzeugs und seiner Kamera in Bezug auf die Fahrbahnmarkierungen zusammen. Die aus dem Bild verschwundenen Kurvenlinien müssen von der Spurerkennung geschätzt werden, was zu Ungenauigkeiten führt. Die Fish-Eye-Kamera zeigt dauerhaft alle Fahrbahnmarkierungen in Kurven.
- Hält ein Fahrzeug an Kreuzungen, kann es nur mit der Fish-Eye-Kamera die abzweigenden Straßen einsehen. Dadurch kann bspw. das Vorfahrtsverhalten geklärt werden. Dies ist mit dem Videobild der Weitwinkel-Kamera nicht möglich.

Zusammenfassend lässt sich sagen, dass es keinen Grund gibt, die Weitwinkel-Kamera der Fish-Eye-Kamera vorzuziehen. Der Fish-Eye-Effekt als einziger Nachteil der entsprechenden Kamera wird durch die neue Spurerkennung neutralisiert, da sie immun gegen diese Art der Bildverzerrung ist. Deshalb wurde entschieden, die Fish-Eye-Kamera als Standardkamera des autonomen Modellfahrzeugs zu verwenden und alle weiteren Videos und Tests mit dieser Kamera durchzuführen.

4.5.2 Anforderungserfüllung der neuen Spurerkennung

Um die Anforderungserfüllung der neuen Spurerkennung zu analysieren, wurden, wie im vorangegangenen Kapitel erwähnt, zunächst Tests auf Videosequenzen durchgeführt. Dabei wurden verschiedene Situationen aufgenommen, die das autonome Modellfahrzeug bspw. beim AADC meistern muss. Dazu gehörten einerseits einfache Rundfahrten auf einem Parcours mit Kurven. Andererseits wurden auch Bildsequenzen betrachtet, die das mehrfache Abbiegen an Kreuzungen oder das Überholen eines Hindernisses darstellten. Alle verwendeten Videos sind im Anhang angefügt.

Anhand der aufgezeichneten Sequenzen konnten bereits erste Erkenntnisse über das Verhalten des Spurerkennungsalgorithmus gesammelt werden. Viel wichtiger war aber, dass mithilfe dieser Videos Anpassungen am Algorithmus (vgl. Kapitel 4.4.3) und seinen Parametern vorgenommen werden konnten.

Nachdem die neue Spurerkennung auf den Bildsequenzen zufriedenstellend funktionierte, wurden außerdem Live-Tests des Algorithmus auf dem selbstkonstruierten, autonomen Modellfahrzeug durchgeführt. Dazu wurde die Spurerkennung auf den Computer des Miniatur-Autos übertragen und das Live-Kamerabild auf dem Fahrzeug durch den Algorithmus analysiert. Im Live-Test kann jede beliebige Situation nachgestellt werden, mit der das Fahrzeug und die Spurerkennung konfrontiert werden könnten.

Beim Live-Test wurde festgestellt, dass der Prozessor des Modellfahrzeugs gerade noch leistungsfähig genug ist, um den Spurerkennungsalgorithmus zu berechnen. Die Spurerkennung musste auf einem in der Auflösung reduzierten Bild (320×240 Pixel) und nur mit drei spatiotemporalen Bildern arbeiten, um den Prozessor nicht voll auszulasten. Reduziert man die Rechenlast durch die Spurerkennung auf ein absolutes Minimum, indem nur ein einziges spatiotemporales Bild genutzt wird, so ist der Prozessor immer noch zu mehr als 75 Prozent ausgelastet. Da allein die Fahrtregelung des Fahrzeugs (vgl. [Fr16]) ca. 30 Prozent des Fahrzeugprozessors veranschlagt, könnten niemals beide Systeme gleichzeitig auf dem autonomen Auto betrieben werden. Ein KI-System, dass die Entscheidungen für das autonome Modellfahrzeug treffen müsste, hätte ebenfalls kaum freie Prozessorkapazitäten. Diese Erkenntnis fließt auch in die Verbesserungsvorschläge für das autonome Modellfahrzeug (Kapitel 3.4) ein.

Anhand der aufgezeichneten Videos und der Live-Tests konnte nun festgestellt werden, welche Anforderungen die neue Spurerkennung ganz, bedingt oder gar nicht erfüllt. Dies wird in Tabelle 2 zusammengefasst. Bedingt erfüllte oder nicht erfüllte Anforderungen werden gesondert im nachfolgenden Kapitel erläutert.

Tabelle 2 – Erfüllung der Anforderungen durch die neue Spurerkennung

Anforderung	Erfüllungsgrad
#1 Durchgezogene Spurmarkierung	Erfüllt
#2 Kreuzungsstabilität	Bedingt Erfüllt
#3 Verdeckungsstabilität	Bedingt Erfüllt
#4 Austauschbarkeit	Erfüllt
#5 Unterbrochene Spurmarkierung	Bedingt Erfüllt
#6 Kameraunabhängigkeit	Erfüllt
#7 Unsicherheitsausgabe	Bedingt Erfüllt
#8 Kreuzungserkennung	Nicht Erfüllt
#9 Parklückenerkennung	Nicht Erfüllt

4.5.3 Grenzen der neuen Spurerkennung

Nicht alle Anforderungen, die an die neue Spurerkennung gestellt wurden, konnten durch den spatiotemporalen Algorithmus vollständig erfüllt werden. Die Gründe dafür sollen nun im Detail erläutert werden.

Bereits bei der Auswahl des spatiotemporalen Algorithmus war klar, dass die neue Spurerkennung keine Kreuzungs- oder Parklückenerkennung umsetzen kann. Dafür ist sie vom theoretischen Ansatz her nicht gedacht und von [JYS16] nicht zu diesem Zweck entworfen worden. Um diese optionalen Anforderungen zu erfüllen, muss ein gesonderter Algorithmus entworfen werden.

Der Algorithmus stößt allgemein an seine Grenzen, wenn das Kamerabild ruckartigen Bewegungen ausgesetzt ist. Dies geschieht bspw., wenn das autonome Modellfahrzeug mit hoher Geschwindigkeit Kurven anfährt oder sich in Schlangenlinien bewegt. Solche Veränderungen im Kamerabild sorgen dafür, dass der spatiotemporale Algorithmus aufeinanderfolgende Bildzeilen stark verschieben muss, um die vertikale Linien im spatiotemporalen Bild zu erhalten. Bei ruckartigen Bewegungen verändert sich das Kamerabild aber zwischen zwei Bildern so stark, dass diese Verschiebung nicht mehr korrekt berechnet werden kann. Es entsteht dadurch eine Abweichung zwischen der aktuellen Bildzeile und den vorangegangenen Bildzeilen, welche für Fehlerkennungen sorgt. Dieser Effekt ist gut in Anhang 6 zu beobachten: Am Ende der Kurve setzt kurz das Kamerabild aus, wodurch ein starker Ruck durchs Videobild geht. Die Spurerkennung ist für ca. drei Sekunden nicht in der Lage, die Spurmarkierungen zu erkennen. Danach funktioniert sie wieder wie erwartet. Mit zunehmendem zeitlichen Abstand zur ruckartigen Bewegung nimmt die Anzahl der Bildzeilen vor der Bewegung ab, wodurch sich die Erkennung langsam selbst korrigiert.

Eine einfache Lösung für dieses Problem scheint darin zu liegen, die Bildzeilenanzahl zu reduzieren, aus der sich das spatiotemporale Bild zusammensetzt. Dadurch wird auch die Zeitspanne der Fehlerkennung nach einer ruckartigen Bewegung reduziert.

Diese Reduktion der Bildzeilenanzahl wirkt sich allerdings auf die Erfüllung anderer Anforderungen aus. Die Bildzeilen vergangener Videobilder dienen dazu, bei verdeckten oder verschwundenen Spurmarkierungen weiterhin abschätzen zu können, wo diese im Augenblick liegen könnten. Wird also die Bildzeilenanzahl reduziert, so sinkt auch die Zeitspanne, in der eine Spurmarkierung geschätzt werden kann. Gut zu erkennen ist dieser Effekt in Anhang 5, in dem das Verhalten an Kreuzungen getestet wird. Nach ca. drei Sekunden Haltezeit an einer Kreuzung kann die Spurerkennung nicht mehr schätzen, wo sich die Mittellinie befinden müsste. Dies liegt daran, dass alle Bildzeilen, die Mittellinieninformationen enthielten, aus dem spatiotemporalen Bild verschwunden sind. Ein ähnliches Verhalten zeigt Anhang 6, bei dem die Linien durch ein Hindernis verdeckt wurden. Da die Haltezeit hier kleiner ist als die Zeitspanne, aus der sich das spatiotemporale Bild zusammensetzt, werden die Spurmarkierungen hier korrekt abgeschätzt.

Aus diesem Grund lassen sich die Anforderungen #2, #3, #5, und #7 nur als bedingt erfüllt betrachten. Prinzipiell kann der Algorithmus unterbrochene Spurmarkierungen verfolgen, Spurpunkte als geschätzt markieren und an Kreuzungen und Verdeckungen stabil bleiben. Je nach Anwendungsfall muss die Anzahl der Bildzeilen allerdings reduziert werden, damit die Spurerkennung auch ruckartige Bewegungen verarbeiten kann. Bei der Einstellung des Algorithmus muss also experimentell eine Bildzeilenanzahl gefunden werden, die so niedrig liegt, dass alle Anforderungen gerade noch erfüllt bleiben. Für die Praxistests mit dem autonomen Modellfahrzeug wurde die Zeitspanne von drei Sekunden bzw. 90 Bildzeilen ermittelt.

Mit dem Ende des Praxistests und dem Aufzeigen der Grenzen der neuen Spurerkennung ist die zweite Hauptaufgabe dieser Arbeit abgeschlossen. Im nachfolgenden Kapitel kann nun eine Schlussbetrachtung erfolgen.

5 Schlussbetrachtung

In Kapitel 1 wurden in der Zielstellung zwei Hauptaufgaben festgelegt. Im Verlauf dieser Arbeit wurde demzufolge dargestellt, wie ein autonomes Modellfahrzeug konzipiert und konstruiert wurde. Im Anschluss wurde für dieses Fahrzeug ein Spurerkennungsalgorithmus entwickelt. Im nun folgenden Fazit sollen die Ergebnisse der Bearbeitung der zwei Hauptaufgaben nochmals kurz zusammengefasst werden. Im Anschluss erfolgt ein Ausblick auf die weiterführende Forschung auf Grundlage dieser Arbeit.

5.1 Fazit

Der erste Hauptteil dieser Arbeit beschäftigte sich mit der Konstruktion eines eigenen Modellfahrzeugs für das Team HTWK Smart-Driving. Das Fahrzeug sollte in der Lage sein, autonome Fahrfunktionen auszuführen und dem Team somit als praktischer Forschungsgegenstand zur Verfügung stehen.

Im Zuge dieser Arbeit wurde ein solches Fahrzeug erfolgreich konstruiert. Es erfüllt alle notwendigen sowie einige optionale Anforderungen, die vor Beginn der Entwicklung festgelegt wurden. Das Modellauto wird bereits erfolgreich vom Team Smart-Driving verwendet: Neben der Entwicklung einer neuen Spurerkennung in dieser Arbeit nutzte [Fr16] in seiner Bachelorarbeit das konstruierte Fahrzeug.

Trotzdem ist das in dieser Arbeit entstandene Fahrzeug als Prototyp zu betrachten. Entsprechend groß ist die Liste an Verbesserungsvorschlägen, die nach den ersten Wochen Forschungsarbeit mit dem Modellauto entstanden ist. Nichts desto trotz spiegelt diese Liste wertvolle Erkenntnisse wider, die im Zuge dieser Arbeit gesammelt werden konnten.

Im Anschluss an die Modellfahrzeugkonstruktion wurde eine Spurerkennung für eben dieses Fahrzeug entwickelt. Der bisher verwendete Spurerkennungsalgorithmus des Teams Smart-Driving offenbarte verschiedene Schwachstellen, die in die Anforderungen an den neuen Algorithmus eingeflossen sind.

Auf Grundlage dieser Anforderungen wurde der theoretische Ansatz von [JYS16] verwendet, um eine neue Spurerkennung zu implementieren. Dabei wurden durch den Autor verschiedene Anpassungen und Verbesserungen des Ansatzes vorgenommen, um den Algorithmus auf das Einsatzgebiet des Teams Smart-Driving abzustimmen.

Die Anforderungserfüllung der neuen Spurerkennung wurde zunächst mithilfe von Videos, später auch durch Live-Tests auf dem selbstkonstruierten Modellfahrzeug überprüft. Außerdem wurden die Grenzen des Algorithmus festgestellt. Es konnte zusätzlich gezeigt werden, dass sich eine Fish-Eye-Kamera am besten als Frontkamera für das Modellfahrzeug eignet.

5.2 Ausblick

Eine Motivation für den Bau des autonomen Modellfahrzeugs war, dass das Team Smart-Driving mit dem Auto in der Lage ist, ganzjährig Forschung auf dem Gebiet des autonomen Fahrens zu betreiben. Mit Fertigstellung des Fahrzeugs konnte dieses Ziel bereits in Angriff genommen werden: Im Augenblick beschäftigt sich das Team mit Hinderniserkennung und digitaler Bildverarbeitung mit künstlichen, neuronalen Netzen und nutzt dabei das autonome Modellfahrzeug. Nebenbei werden die bereits existierenden Module des autonomen Fahrsystems für den AADC von ADTF auf ROS portiert. Dadurch können diese bald auch auf dem neuen Modellfahrzeug verwendet werden.

Zusätzlich beschäftigt sich ein Teil des Teams Smart-Driving damit, eine zweite, verbesserte Version des Modellfahrzeugs zu konstruieren. Dabei fließt die Liste an Verbesserungsvorschlägen aus dieser Arbeit als Grundlage in den Entwurf ein. Ein kurzfristiges Ziel ist dabei, das zweite Modellfahrzeug bereits für den Carolo Cup 2017 fertigzustellen.

Für diesen Wettbewerb kann auch die in dieser Arbeit entwickelte Spurerkennung verwendet werden. Dazu befindet sich im Augenblick ein Fahrtenregler in Entwicklung, der die im Video erkannten Spurpunkte in Lenkansteuerungen umrechnet. Dadurch wird das Fahrzeug auf der Fahrspur gehalten. Mit Fertigstellung des Fahrtenreglers kann dieser gemeinsam mit dem Spurerkennungsalgorithmus für die anstehenden Wettbewerbe genutzt werden.

Alles in allem lässt sich festhalten, dass mit dieser Arbeit neue Forschungsmöglichkeiten für das Team Smart-Driving geschaffen wurden. Auf den vorgestellten Ergebnissen kann aufgebaut und ein Anreiz für neue Teammitglieder geschaffen werden. Dadurch steht der Zukunft des autonomen Fahrens an der HTWK aus technischer Sicht nichts mehr im Weg.

Literaturverzeichnis

[Au15] Audi AG: Audi Autonomous Driving Cup Manual, 2015.

[BBF98] Bertozz, M.; Broggi, A.; Fascioli, A.: Stereo inverse perspective mapping: theory and applications. In Image and Vision Computing, 1998, 16; S. 585–590.

[BK08] Bradski, G. R.; Kaehler, A.: Learning OpenCV. Computer vision with the OpenCV library. O'Reilly, Sebastopol, Calif., 2008.

[CSS14] Chaki, N.; Shaikh, S. H.; Saeed, K.: Exploring image binarization techniques. Springer, New Delhi, 2014.

[Da06] Daf-de: Hough Transform, example results. https://commons.wikimedia.org/wiki/File:Hough-example-result.png.

[Du14] Duden: Duden | au-to-nom | Rechtschreibung, Bedeutung, Definition, Synonyme, Herkunft. http://www.duden.de/node/650163/revisions/1617396/view, 07.08.2016.

[Fa90] Faugeras, O. Hrsg.: Computer vision - ECCV 90. First European Conference on Computer Vision, Antibes, France, April 23 - 27, 1990 ; proceedings. Springer, Berlin, 1990.

[Fr16] Freihube, F.: Entwurf und Realisierung autonomer Fahrfunktionen in Modellfahrzeugen. Bachelorarbeit, Leipzig, 2016.

[GW08] Gonzalez, R. C.; Woods, R. E.: Digital image processing. Pearson Education Internat, Upper Saddle River, NJ, 2008.

[Ha15] Hamburg Port Authority: Pressemitteilung - Staufrei durch den Hamburger Hafen: HPA und NXP stellen intelligente Ampel vor. http://www.hamburg-port-authority.de/de/presse/pressearchiv/Seiten/Pressemitteilung-28-05-2015.aspx, 26.08.2016.

[HLN12] Hertzberg, J.; Lingemann, K.; Nüchter, A.: Mobile Roboter: Eine Einführung aus Sicht der Informatik. Springer Berlin Heidelberg, 2012.

[Hu06] Hua-jun, L. et al.: A Fast Method for Vanishing Point Estimation and Tracking and Its Application in Road Images. In (Wen, G. Hrsg.): 2006 6th International Conference on ITS Telecommunications proceedings. June 2006, [Chengdu, China]. IEEE Operations Center, Piscataway, NJ, 2006; S. 106–109.

[Jä05] Jähne, B.: Digitale Bildverarbeitung. Springer-Verlag Berlin Heidelberg, Berlin, Heidelberg, 2005.

[JM15] Johanning, V.; Mildner, R.: Car IT kompakt. Das Auto der Zukunft - Vernetzt und autonom fahren. Springer Vieweg, Wiesbaden, 2015.

[JYS16] Jung, S.; Youn, J.; Sull, S.: Efficient Lane Detection Based on Spatio-temporal Images. In IEEE Transactions on Intelligent Transportation Systems, 2016, 17; S. 289–295.

[Op16] OpenCV: OpenCL. http://opencv.org/platforms/opencl.html, 28.10.2016.

[Se01] Seal, D. Hrsg.: ARM architecture reference manual. Addison-Wesley, Harlow, 2001.

[UHF13] Upton, E.; Halfacree, G.; Feilen, M.: Raspberry Pi. Einstieg und User Guide. mitp, Heidelberg, Hamburg, 2013.

[VG04] Vintar, J.; Goldsman, A.: I, Robot, 2004.

[Vi15] Vieweg, C.: Wer hat das Roboterauto erfunden? Die Bundeswehr! Autonomes Fahren. http://www.zeit.de/mobilitaet/2015-07/autonomes-fahren-geschichte, 05.08.2016.

[We14] Weber, M.: Where to? A History of Autonomous Vehicles. http://www.computerhistory.org/atchm/where-to-a-history-of-autonomous-vehicles/, 05.08.2016.

[YCC95] Yen, J. C.; Chang, F. J.; Chang, S.: A new criterion for automatic multilevel thresholding. In IEEE transactions on image processing a publication of the IEEE Signal Processing Society, 1995, 4; S. 370–378.

Anhangsverzeichnis

Anhang 1 Video: Äußere Umrundung mit Fish-Eye-Kamera

Dieses Video zeigt die Umrundung eines Parcours durch das autonome Modellfahrzeug, das mit einer Fish-Eye-Kamera ausgestattet ist. Der Parcours besteht aus einer T-Kreuzung mit rechtem Abzweig, einer 180°-Grad-Kurve und einer T-Kreuzung mit linkem Abzweig. Kurz vor Kurveneingang fehlt die mittlere, gestreifte Spurmarkierung auf einer Strecke von ca. 1,5 Metern. Das Fahrzeug bewegt sich auf der äußeren Spur.

Dieses Video ist als digitaler Anhang auf der beiliegenden CD enthalten.

Anhang 2 Video: Innere Umrundung mit Fish-Eye-Kamera

Dieses Video zeigt die Umrundung eines Parcours durch das autonome Modellfahrzeug, das mit einer Fish-Eye-Kamera ausgestattet ist. Der Parcours besteht aus einer T-Kreuzung mit rechtem Abzweig, einer 180°-Grad-Kurve und einer T-Kreuzung mit linkem Abzweig. Kurz nach dem Kurvenausgang fehlt die mittlere, gestreifte Spurmarkierung auf einer Strecke von ca. 1,5 Metern. Das Fahrzeug bewegt sich auf der inneren Spur.

Dieses Video ist als digitaler Anhang auf der beiliegenden CD enthalten.

Anhang 3 Video: Äußere Umrundung mit Weitwinkel-Kamera

Dieses Video zeigt die Umrundung eines Parcours durch das autonome Modellfahrzeug, das mit einer Weitwinkel-Kamera ausgestattet ist. Der Parcours besteht aus einer T-Kreuzung mit rechtem Abzweig, einer 180°-Grad-Kurve und einer T-Kreuzung mit linkem Abzweig. Kurz vor Kurveneingang fehlt die mittlere, gestreifte Spurmarkierung auf einer Strecke von ca. 1,5 Metern. Das Fahrzeug bewegt sich auf der äußeren Spur.

Dieses Video ist als digitaler Anhang auf der beiliegenden CD enthalten.

Anhang 4 Video: Innere Umrundung mit Weitwinkel-Kamera

Dieses Video zeigt die Umrundung eines Parcours durch das autonome Modellfahrzeug, das mit einer Weitwinkel -Kamera ausgestattet ist. Der Parcours besteht aus einer T-Kreuzung mit rechtem Abzweig, einer 180°-Grad-Kurve und einer T-Kreuzung mit linkem Abzweig. Kurz nach dem Kurvenausgang fehlt die mittlere, gestreifte Spurmarkierung auf einer Strecke von ca. 1,5 Metern. Das Fahrzeug bewegt sich auf der inneren Spur.

Dieses Video ist als digitaler Anhang auf der beiliegenden CD enthalten.

Anhang 5 Video: Kreuzungsfahrt mit Fish-Eye-Kamera

Dieses Video zeigt eine Kreuzungsfahrt des autonomen Modellfahrzeugs, das mit einer Fish-Eye-Kamera ausgestattet ist. Die Fahrt beginnt mit einer T-Kreuzung mit linkem Abzweig, an der nach einer Sieben-Sekunden-Pause links abgebogen wird. Nach zwei Metern Fahrt folgt erneut eine T-Kreuzung mit linkem und rechtem Abzweig. Nach einer Sieben-Sekunden-Pause wird hier rechts abgebogen. Nach einer 180°-Grad-Kurve, an deren Ausgang die mittlere, gestreifte Spurmarkierung auf einer Strecke von ca. 1,5 Metern fehlt, steht das Fahrzeug an einer T-Kreuzung mit rechtem Abzweig. Nach einer Fünf-Sekunden-Pause überquert das Fahrzeug die Kreuzung geradeaus.

Dieses Video ist als digitaler Anhang auf der beiliegenden CD enthalten.

Anhang 6 Video: Hindernis und Spurwechsel mit Fish-Eye-Kamera

Dieses Video entspricht im Wesentlichen Anhang 1. Vor der ersten Kreuzung trifft das Fahrzeug allerdings auf ein Hindernis. Es hält vor dem Hindernis und wartet fünf Sekunden. Danach setzt es kurz zurück und umfährt das Hindernis auf der Gegenfahrbahn, ehe es wieder auf seine Fahrbahn einbiegt.

Dieses Video ist als digitaler Anhang auf der beiliegenden CD enthalten.

Eidesstattliche Versicherung

Name:	Jenschmischek	Vorname:	Georg
Matrikel-Nr.:	64572	Studiengang:	Informatik Master

Hiermit versichere ich, Georg Jenschmischek, an Eides statt, dass ich die vorliegende Masterarbeit mit dem Titel „Digitale Bildverarbeitung auf einem selbst konstruierten Modellfahrzeug“ selbständig und ohne fremde Hilfe verfasst und keine anderen als die angegebenen Hilfsmittel benutzt habe. Die Stellen der Arbeit, die dem Wortlaut oder dem Sinne nach anderen Werken entnommen wurden, sind in jedem Fall unter Angabe der Quelle kenntlich gemacht. Die Arbeit ist noch nicht veröffentlicht oder in anderer Form als Prüfungsleistung vorgelegt worden.

Ich habe die Bedeutung der eidesstattlichen Versicherung und prüfungsrechtlichen Folgen sowie die strafrechtlichen Folgen einer unrichtigen oder unvollständigen eidesstattlichen Versicherung zur Kenntnis genommen.

Auszug aus dem Strafgesetzbuch (StGB)

§ 156 StGB Falsche Versicherung an Eides Statt

Wer von einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

Leipzig, 09. Dezember 2016

Georg Jenschmischek