Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Masterarbeit

Eike Jenning

Systemidentifikation eines autonomen Fahrzeugs mit einer robusten, kamerabasierten Fahrspurerkennung in Echtzeit

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Eike Jenning

Systemidentifikation eines autonomen Fahrzeugs mit einer robusten, kamerabasierten Fahrspurerkennung in Echtzeit

Masterarbeit eingereicht im Rahmen der Masterprüfung
im Studiengang Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. rer. nat. Stephan Pareigis
Zweitgutachter: Prof. Dr.-Ing. Andreas Meisel

Abgegeben am 05.12.2008

Eike Jenning

Thema der Masterarbeit

Systemidentifikation eines autonomen Fahrzeugs mit einer robusten, kamerabasierten Fahrspurerkennung in Echtzeit

Stichworte

FAUST, Fahrspurerkennung, Bildverarbeitung, Echtzeit, Systemidentifikation, Ausgleichsrechnung, Linsenverzeichnungskorrektur, projektive Transformatione, autonomes Fahrzeug, Carolo-Cup

Kurzzusammenfassung

Zur Teilnahme am Carolo-Cup der Technischen Universität Braunschweig wird im Forschungsprojekt FAUST der Hochschule für Angewandte Wissenschaften Hamburg ein autonomes Modellfahrzeug mit komplexen Regelungs- und Kartierungssystemen ausgestattet. In dieser Arbeit wird ein Fahrspurerkennungsverfahren entwickelt, das den genannten Systemen als Informationsgrundlage dient. Zu diesem Zweck wird die Fahrspur durch ein kubisches Polynom approximiert und anschließend die Fahrzeuglage in der Fahrspur, der tangentiale Steuerkurswinkel des Fahrzeugs zur Fahrspur und der Kurvenradius der Fahrspur identifiziert.

Title of the paper

Realtime system identification of an autonomous vehicle using robust camera-based lane detection

Keywords

FAUST, lane detection, image processing, realtime, system identification, curve fitting, lens distortion correction, projective transformation, autonomous vehicle, Carolo-Cup

Abstract

In the research project FAUST at the University of Applied Sciences Hamburg an autonomes model car will be equipped with complex control and mapping systems for the participation at the Carolo-Cup of the Technical University Braunschweig. This thesis describes the development of a lane detection system that provides vital information to the control and mapping systems. Therefore the lane is approximated with a third order polynomial. Consequently the polynomial is used to identify the lateral position and the tangential heading angle of the model car to the lane as well as the radius of curvature of the lane.

Danksagung

An dieser Stelle halte ich es für angebracht mich bei einigen Personen, die mich während des Studiums und bei der Anfertigung dieser Arbeit begleitet und unterstützt haben, zu bedanken.

Allen voran danke ich meinem Betreuer Stephan Pareigis, der mich schon seit dem Bachelor-Studium begleitet. Immer wenn es langweilig zu werden drohte, bot er spannende Aufgaben und Perspektiven. Vermutlich ist er sich gar nicht darüber bewusst, wie sehr er meine Entwicklung positiv beeinflusst hat. Ohne ihn wäre ich heute ein anderer Mensch.

Auch meinem Zweitprüfer Herrn Meisel möchte ich an dieser Stelle für die intensiven Gespräche während der Anfertigung dieser Arbeit danken. Die Fülle an Ideen muss man erstmal verarbeiten...

Mit Enrico Hensel, habe ich einen tollen Kollegen und Freund kennengelernt, mit dem ich viel lachen und schaffen konnte. Dafür danke ich ihm sehr.

Hinten in unserer „Masterecke“ im 7. Stock saß ich mit zwei Frohnaturen, Dirk Ewerlin und Nico Manske, zusammen, mit denen man selbst stressige Zeiten prima durchstehen kann. Vom Fachgespräch zum Lachgespräch gings oft innerhalb einer Sekunde. Kaffee Junge, echt jetzt.

Carsten Schulz gesellte sich sporadisch ebenfalls zu unserer „Masterecke“ hinzu. Auch bei ihm möchte ich mich für die tolle gemeinsame Masterzeit in der Hochschule und außerhalb bedanken. Es hat Spaß gemacht mit dir zu lernen und zu feiern.

Sebastian Gregor war wieder einmal eine verlässliche Hilfe bei der Korrektur dieser Arbeit und sorgte in anstrengenden Zeiten für willkommene Ablenkung durch gemeinsames Fußballschauen.

Auch den anderen Korrekturlesern Christine Jakstat und Lieselotte Jenning danke ich für die tatkräftige Unterstützung.

Zuletzt möchte ich Kaja danken, weil sie schon so lange mein tolles Mädchen ist.

Inhaltsverzeichnis

1 Einleitung

Das Projekt FAUST, kurz für Fahrerassistenz- und Autonome Systeme, aus dem Department für Informatik der Hochschule für Angewandte Wissenschaften (HAW) Hamburg verfolgt die Analyse und Synthese von verteilten, eingebetteten Echtzeitsystemen. Zu diesem Zweck werden (Modell-)Fahrzeuge mit Sensorik, Aktorik und Steuereinheiten ausgestattet, um beispielsweise Ausweichassistenten, Stabilisierungsverfahren oder zeitgesteuerte Systeme zu entwickeln und zu erproben.

Aufgrund des stetigen Erfahrungszuwachses im Umgang mit autonomen und Fahrerassistenzsystemen steigen auch die Anforderungen an den Leistungsumfang dieser Systeme. Eine der aktuellen Anforderungen ist die Implementierung von Bildsensorverarbeitungen. Beispielsweise werden Landmarken zur groben Orientierung der Fahrzeuge in Außenbereichen [Tarik Mouslih (2007)] oder Reflektormarken zur zentimetergenauen Positionsbestimmung in Innenbereichen [Nico Manske (2008)] durch bildverarbeitende Systeme ausgewertet. Zudem werden aktuell Lösungen zur Erkennung von Fahrspuren für die Navigation von autonomen Fahrzeugen entwickelt [Alexander Kant (2007), Dennis Berger (2008a), Dennis Berger (2008b), Nils Kruse (2008)].

In der aktuellen Forschungsliteratur werden dynamische und statische Verfahren zur Fahrspurerkennung unterschieden. Ziel der Verfahren ist jeweils die Bestimmung der Parameter eines Fahrspurmodells zur Approximation der Fahrspurmarkierungen.

Durch den Einsatz von mehreren Differentialgleichungssystemen berechnen die dynamischen Verfahren unter Berücksichtigung des zeitlichen Verlaufs der Fahrspur die Parameter des Fahrspurmodells für das aktuelle Bild. Die Berücksichtigung des zeitlichen Verlaufs ermöglicht eine effektive Ausgleichung von Störungen bei der Parameterbestimmung durch Anwendung eines Kalman-Filters. Dieser Ansatz wird in der Literatur als „4d-Ansatz“ [Ernst D. Dickmanns und Birger D. Mysliwetz (1992), Stephan Neumaier und Georg Färber (2005)] oder „lane tracking“ [Keith A. Redmill u. a. (2001)] bezeichnet . Variationen dieses Ansatzes sind unter anderem die Verwendung von je einem dynamischen Fahrspurmodell für den nahen und fernen Bildbereich [Deepak Khosla (2002)], die Ergänzung um eine Erkennung des optischen Flusses [Axel Gern u. a. (2002)], oder die Vorgabe eines konstanten Störprozesses zum Verzicht auf den Kalman-Filter [R. Risack u. a. (1998)].

Gegenstand dieser Arbeit ist die Verwendung eines statischen Verfahrens zur Fahrspurerkennung. Im Gegensatz zu dynamischen Verfahren wird demnach der zeitliche Verlauf der Fahrspur nicht berücksichtigt, sodass keine komplexen Differentialgleichungssysteme aufgestellt werden müssen. Stattdessen werden die Parameter eines Fahrspurmodells jeweils nur anhand des aktuellen Bildes bestimmt. Als Fahrspurmodell wird ein Polynom 3. Grades

$$p(x) = ax^3 + bx^2 + cx + d \tag{1.1}$$

eingesetzt und zur Literatur über statische Verfahren in Bezug gestellt (vgl. Abschnitt 2.3). Ergänzend kommt eine Vorhersage für die Parameter des Polynoms im Folgebild zum Einsatz (vgl. Abschnitt 2.4). Die sukzessive Erweiterung des statischen Verfahrens um dynamische Komponenten ist damit vorbereitet (vgl. Abschnitt 7.3).

Nachfolgend wird zunächst die Motivation dieser Arbeit in Abschnitt 1.1 dargestellt. Abschnitt 1.2 beschreibt anschließend das Szenario, in dem die Konzepte dieser Arbeit entwickelt werden. Eine Zielvereinbarung in Abschnitt 1.3 dient der Arbeit als Qualitätsrichtlinie. Abschnitt 1.4 stellt in einer Übersicht das entwickelte Verfahren dar und präsentiert den konsequenten Aufbau dieser Arbeit.

1.1 Motivation dieser Arbeit

Im Februar 2008 fand ausgetragen von der Technischen Universität in Braunschweig erstmalig der jährliche Carolo-Cup statt [Carolo-Cup]. Bei diesem Wettkampf treten Studierendenteams verschiedener Hochschulen mit autonom fahrenden Modellfahrzeugen in den Disziplinen *Rundstreckenfahrt ohne Hindernisse*, *Rundstreckenfahrt mit Hindernissen* und *Einparken* gegeneinander an [Carolo-Cup-Regelwerk (2009)]. Eine große Herausforderung ergibt sich dabei aus der geregelten Spurführung des Fahrzeugs. Diese setzt sich aus einer Verarbeitungskette bestehend aus

- einer Fahrspurerkennung zur Erzeugung der Fahrspurinformationen,
- einer Hindernisserkennung zur Erzeugung von Hindernisinformationen,
- einer Regelungssoftware zur Spurhaltung,
- und einer Regelungssoftware zum Spurwechsel beim Ausweichen

zusammen. Die Daten der Fahrspurerkennung sind demnach ein wesentlicher Faktor für den Erfolg der Spurführung.

Für die Teilnahme am Carolo-Cup 2008 wurde die Fahrspurerkennung *TFALDA* [Young Uk Yim und Se-Young Oh (2003)] nach einer vorausgehenden Recherche [Eike Jenning (2008b)] ausgewählt und implementiert [Dennis Berger (2008a), Eike Jenning (2008a)]. Das zu diesem Zeitpunkt bereits entwickelte Verfahren zur Fahrspurerkennung basiert auf einer Hough-Transformation zur Detektion von Geraden [Alexander Kant (2007)] und ist deshalb nicht auf die Anforderungen des Wettbewerbs [Carolo-Cup-Regelwerk (2009)] übertragbar. Die *TFALDA* Fahrspurerkennung wurde nach dem Wettbewerb weiter verbessert [Dennis Berger (2008b)]. Der aktuelle Stand wird in dieser Arbeit kurz dargestellt und diskutiert (vgl. Abschnitt 3.1.1).

Die Fahrspurinformationen werden durch *TFALDA* in Form einer Punktekolonne für die linke und rechte Fahrspurmarkierung zur Verfügung gestellt [Eike Jenning (2008a), Abbildung 2.5]. Die Punktekolonne gibt die Position der Fahrspurmarkierung im Bild an, wodurch die Punkte in Richtung eines Fluchtpunkts angeordnet sind. Dadurch erschwert sich dessen Interpretation für die Regelung, da kleine Veränderungen der Punkte nahe des Fluchtpunkts aufgrund der Entfernung zur Kamera große Veränderungen der Fahrspurmarkierung abbilden.

Der zur Teilnahme am Carolo-Cup eingesetzte Reglerentwurf wird in [Enrico Hensel (2008), Abschnitt 6] kurz beschrieben und konzeptionell erweitert. Das neue Konzept basiert auf einer Zusammenfassung verschiedener Systemgrößen zur Erhöhung der Zuverlässigkeit des Reglers. So lässt sich beispielsweise die Lenkung in Abhängigkeit der aktuellen Geschwindigkeit einstellen.

Diese Arbeit dient der Unterstützung des neuen Reglerentwurfs durch eine konsequente Optimierung der Fahrspurerkennung. Die Einführung des Fahrzeugkoordinatensystems (vgl. Abschnitt 2.2) erlaubt die Darstellung der Fahrspur, wie sie tatsächlich vor dem Fahrzeug verläuft. Dadurch lassen sich seitliche Abstände auf einen Zentimeter genau regeln. Es werden Verfahren zur Transformation von Bildpunkten in das Fahrzeugkoordinatensystem (vgl. Abschnitt 3.3) und zur zusätzlichen Verbesserung der Genauigkeit (vgl. 3.2) eingeführt.

Erweitert wird die Fahrspurerkennung durch ein Systemidentifikationsmodul, das die Lage und Orientierung des Fahrzeugs zur Fahrspur mit Hilfe des mathematischen Fahrspurmodells berechnet (vgl. Kapitel 4). Zusätzlich wird durch die Systemidentifikation der Radius der Fahrspurmarkierung bezogen auf eine vorgegebene Position im Fahrzeugkoordinatensystem extrahiert.

1.2 Anwendungsszenario zur Erprobung der Konzepte

Das Szenario zur Erprobung der Konzepte dieser Arbeit folgt aus der Motivation, den Reglerentwurf und somit auch die Fahrspurerkennung für die Teilnahme am nächsten Carolo-Cup konzeptionell zu verbessern.

Ein Modellfahrzeug wird für die Bildverarbeitung mit einem Hauptrechner und einer Kamera ausgestattet. Das Fahrzeug bewegt sich auf einer Fahrbahn mit definierten Eigenschaften [Carolo-Cup-Regelwerk (2009)]:

- Fahrspurmarkierungen sind weiß und haben eine definierte, konstante Breite,
- Fahrspuren haben eine definierte, konstante Breite,
- die Fahrbahn besteht aus zwei Fahrspuren, getrennt durch eine Leitlinie,
- die Leitlinie hat ein definiertes, konstantes Intervall

Die Fahrbahn besteht aus Geraden unbekannter Länge, Kurven mit festgelegtem, minimalen Kurveninnenradius und Kreuzungen definierter Länge, die geradeaus durchfahren werden. Die Kamera zeichnet in engen Kurven mindestens die beiden Markierungen der befahrenen Fahrspur auf.

Die Fahrbahn befindet sich zudem in einer Ebene. Sie weist keine Höhenunterschiede auf. Die Kamera ist in einem konstanten Winkel zur Fahrbahn angebracht, sodass das Verhältnis der Bildebene zur Fahrbahnebene stets identisch ist.

1.3 Zielvereinbarungen für diese Arbeit

Folgende Zielvereinbarungen werden für die Konzepte der Fahrspurerkennung getroffen:

- Die Fahrspurerkennung soll eine Wiederholrate von mindestens 60 Bildern pro Sekunde erreichen.

- Die Fahrspurerkennung muss gegenüber seitlichem Lichteinfall oder ähnlichen Bildstörungen robust sein. Des Weiteren muss das Verfahren gemäß Szenario (vgl. Abschnitt 1.2) gegenüber Kreuzungen und der Leitlinie unempfindlich sein.

- Die Fahrspurerkennung soll Geraden und Kurven gleichermaßen gut unterstützen. Zudem wird eine gute Verarbeitung von S-Kurven und Fahrspurwechseln gefordert.

- Die Fahrspurerkennung soll die Fahrspurinformationen im metrischen Fahrzeugkoordinatensystem (vgl. Abschnitt 2.2) mindestens auf einen Zentimeter genau bereit halten.

Die Systemidentifikation stellt wichtige Parameter für die Regelung des Fahrzeugs [Enrico Hensel (2008)] und die Kartographierung der Fahrzeugumgebung [Andrej Rull (2008), Michael Ebert (2008)] bereit. Folgende Parameter sollen extrahiert werden:

- Der laterale Abstand A_h in $[mm]$ des Fahrzeugs zur Mitte der Fahrspur. Ein positiver Abstand soll eine Abweichung nach rechts und ein negativer Abstand eine Abweichung nach links signalisieren.

- Der tangentiale Steuerkurswinkel ψ in $[deg]$ des Fahrzeugs zu einer Fahrspurmarkierung. ψ gibt an, ob das Fahrzeug parallel zur Fahrspur fährt. Ein positiver Steuerkurswinkel soll eine Abweichung nach rechts, ein negativer Steuerkurswinkel eine Abweichung nach links signalisieren.

- Der Radius der Fahrspur r in [mm]. Der Radius einer Geraden ist durch eine entsprechend große Zahl zu signalisieren. Ein positiver Radius soll eine Kurve nach rechts, und ein negativer Radius eine Kurve nach links signalisieren.

1.4 Übersicht der Verfahrensschritte und konsequenter Aufbau dieser Arbeit

Einleitend in diese Arbeit wird in Abbildung 1.1 auf Seite 6 eine Übersicht über die einzelnen Verfahrensschritte zur Fahrspurerkennung und Systemidentifikation dargestellt. Der Aufbau dieser Arbeit folgt zur Verbesserung der Übersicht der Reihenfolge der Verfahrensschritte.

Vor der Beschreibung der Fahrspurerkennung wird in **Kapitel 2** das mathematische Fahrspurmodell eingeführt. Zusätzlich werden das verwendete Koordinatensystem und eine Vorhersage der Modellparameter für das Folgebild beschrieben.

Die Verfahrensschritte 1-3 in Abbildung 1.1 sind eine Veranschaulichung der entsprechenden Verfahrensschritte der Fahrspurmodellapproximation in **Kapitel 3**. Zunächst werden zur Fahrspurerkennung Betrachtungsbereiche im Bild platziert und in diesen die Fahrspurmarkierung ermittelt (vgl. Abbildung 1.1, oben links). Jeder Betrachtungsbereich extrahiert maximal einen Punkt, der die Lage der Fahrspurmarkierung im Bild beschreibt. Die anschließende Linsenverzeichnungskorrektur berechnet die Pixelkoordinaten der extrahierten Punkte in einem verzeichnungsfreien Bild (vgl. Abbildung 1.1, oben rechts). Die projektive Transformation bestimmt daraufhin die Positionen der verzeichnungsfreien Bildpunkte im Fahrzeugkoordinatensystem anhand kalibrierter Transformationsparameter (vgl. Abbildung 1.1, mitte rechts). Sobald die Punkte im Fahrzeugkoordinatensystem vorliegen, werden die Parameter des linken und rechten Polynoms zur Abbildung der Fahrspurmarkierungen durch eine Ausgleichsrechnung approximiert (vgl. Abbildung 1.1, mitte links). Die Ausrichtung der Betrachtungsbereiche erfolgt im Fahrzeugkoordinatensystem und muss auf das Bild übertragen werden. Die beschriebene Kette wird dazu rückgängig durchlaufen.

In **Kapitel 4** wird die Systemidentifikation der Fahrzeuglage, des Fahrzeugsteuerkurses und des Fahrspurradius beschrieben (vgl. Abbildung 1.1, unten links). Die Identifikation wertet die approximierten Polynome aus und stellt die Ergebnisse zur optionalen Verwendung in Regelungs- und Kartographierungssystemen bereit 1.1, unten rechts).

Eine messtechnische Auswertung der Approximations- und Identifikationsschritte wird in **Kapitel 5** durchgeführt. Neben den Approximationseigenschaften des gewählten kubischen Polynoms und dessen Auswirkungen auf die Identifikation werden die Testumgebung beschrieben und die Ausführungszeiten der Verfahrensschritte gemessen.

Kapitel 6 fasst die Inhalte dieser Arbeit und den Stand des Verfahrens zusammen. Ein Ausblick auf weitere Entwicklungsschritte rundet diese Arbeit in **Kapitel 7** ab.

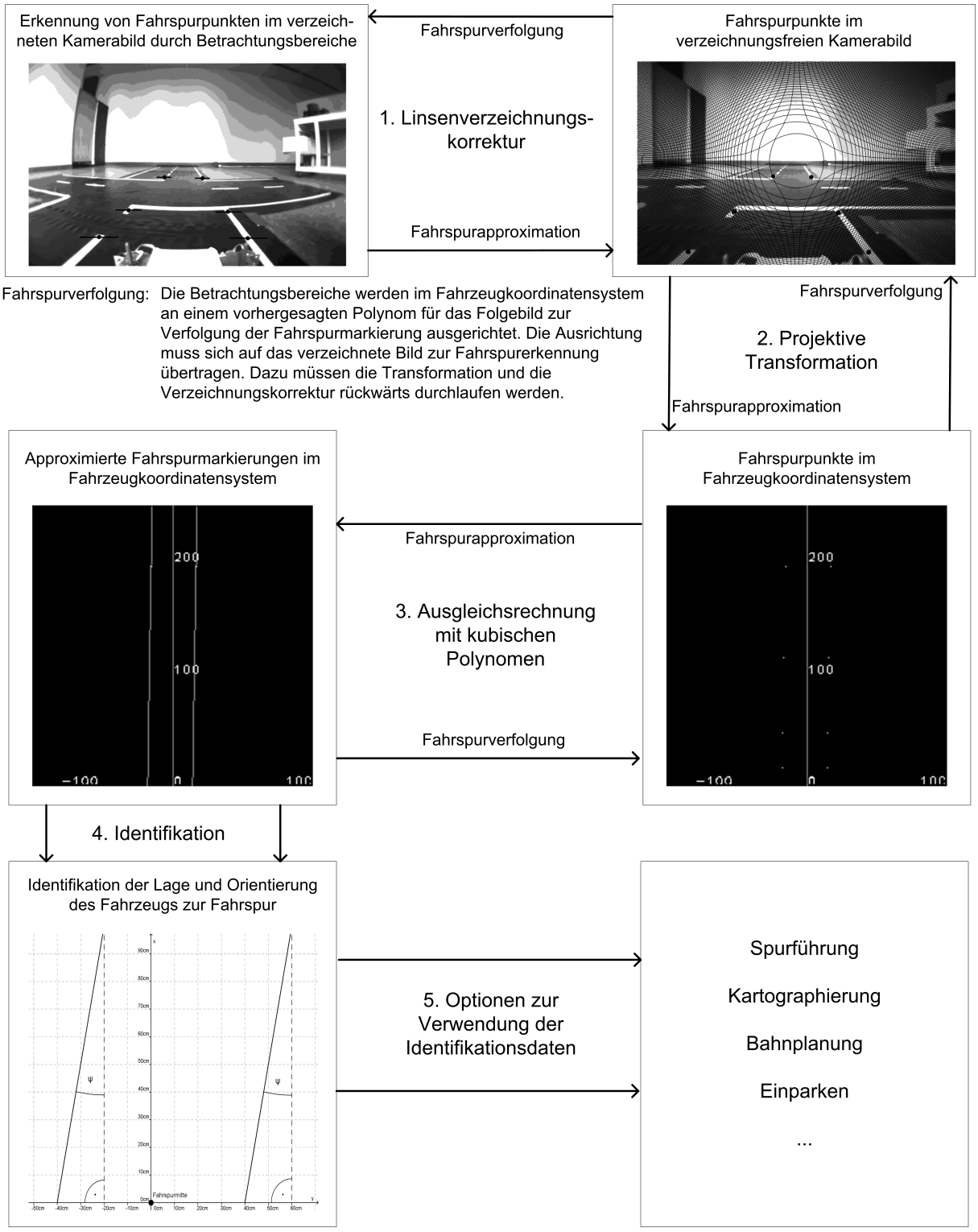


Abb. 1.1: Übersicht der Verfahrensschritte zur Approximation der Fahrspur mit anschließender Identifikation der Fahrzeuglage und Fahrzeugorientierung

2 Mathematisches Fahrspurmodell zur Abbildung von Fahrspurmarkierungen

Das mathematische Fahrspurmodell dient zur Approximation der linken und rechten Fahrspurmarkierungen anhand vereinzelter Fahrspurmesswerte (vgl. Abschnitt 3.1). Die Identifikation der Fahrzeuglage und des Fahrzeugkurses basieren auf der Auswertung des Fahrspurmodells (vgl. Kapitel 4).

Zunächst werden die Anforderungen an das mathematische Fahrspurmodell spezifiziert (vgl. Abschnitt 2.1) und das verwendete Koordinatensystem eingeführt (vgl. Abschnitt 2.2). Anschließend wird in Abschnitt 2.3 ein statisches, mathematisches Fahrspurmodell der Forschungsliteratur dargestellt und die Überführung in ein kubisches Polynom begründet. Abschließend wird in Abschnitt 2.4 eine Methode zur Vorhersage der Polynomparameter für das Folgebild präsentiert.

2.1 Spezifikation der Anforderungen an das Fahrspurmodell

Die Anforderungen an das Fahrspurmodell folgen aus den Zielvereinbarungen für diese Arbeit (vgl. Abschnitt 1.3) und orientieren sich an dem Carolo-Cup Regelwerk [Carolo-Cup-Regelwerk (2009)]. Nachfolgend werden die Anforderungen aufgelistet und spezifiziert:

- **Stetigkeit**: Das Fahrspurmodell soll die Fahrspurmarkierungen stetig im Raum approximieren. Diese Anforderung folgt aus der Geschlossenheit der Carolo-Cup Wettkampfstrecke. Lücken in den Fahrspurmarkierungen deuten entweder auf eine Mittellinie oder auf eine geradeaus zu durchquerende Kreuzung hin. In beiden Fällen ist die stetige Fortsetzung der erkennbaren Fahrspurmarkierung die richtige Lösung.

- **Freiheitsgrade**: Das Fahrspurmodell soll ausreichend Freiheitsgrade zur Approximation von Geraden und einfachen Kurven besitzen. Die Anzahl der Freiheitsgrade soll das Fahrspurmodell jedoch nicht beliebig flexibel gestalten. Andernfalls passt sich das Modell Messwertfehlern zu stark an.

- **Komplexität**: Das Fahrspurmodell soll möglichst einfach aufgebaut sein, um die Komplexität der Approximation zu verringern. Konsequent würden sich die Umsetzung der Approximation und die Analyse der Verfahrensschritte erleichtern(vgl. Abbildung 1.1). Nach der Erprobung der Verfahrensschritte ließe sich die Komplexität des Fahrspurmodells sukzessive erhöhen.

2.2 Auswahl des Koordinatensystems

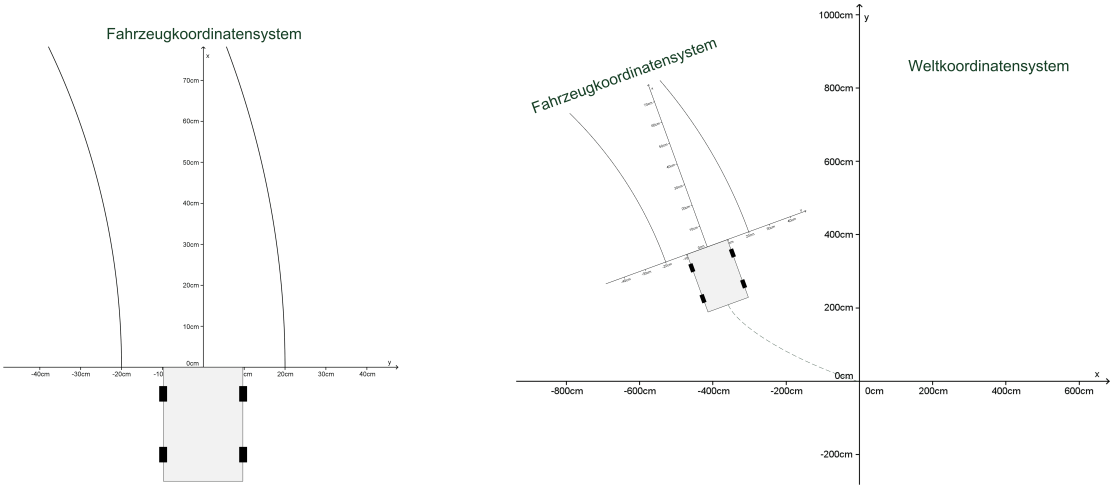
Zur Approximation des Fahrspurmodells wird ein Koordinatensystem benötigt, in dem die Messwerte angeordnet werden. Das Szenario (vgl. Abschnitt 1.2) impliziert die Lage des Koordinatensystems entweder in der Bildebene oder in der Weltebene. Durch das Aufspannen des Koordinatensystems in der Weltebene erlangt die Fahrzeugsteuerung genaue Kenntnis über die Lage und den Steuerkurs des Fahrzeugs bezogen auf die Fahrspur (vgl. Kapitel 4). Das Koordinatensystem liegt aus diesem Grund in der Weltebene.

Grundsätzlich lassen sich zwei Handhabungen des Koordinatensystems in der Weltebene unterscheiden in:

1. **Fahrzeugkoordinatensystem** (vgl. Abbildung 2.1a): Ein definierter Punkt des Fahrzeugs liegt starr im Koordinatenursprung. Das Koordinatensystem wird mit dem Fahrzeug bewegt. Alle Informationen im Koordinatensystem stellen den Auschnitt der Welt dar, den das Fahrzeug zu einem Zeitpunkt wahrnimmt.

2. **Weltkoordinatensystem** (vgl. Abbildung 2.1b): Das Fahrzeug wird frei beweglich in das Koordinatensystem eingezeichnet. Das Koordinatensystem wird zu keiner Zeit bewegt. Informationen werden im Weltkoordinatensystem gespeichert.

In der Kartographierung wird das Fahrzeugkoordinatensystem als lokale Karte [Michael Ebert (2008)] und das Weltkoordinatensystem als globale Karte [Andrej Rull (2008)] bezeichnet. Die Messungen der Sensoren werden in separaten lokalen Karten abgelegt. Die lokalen Karten werden fusioniert und anschließend in die globale Karte integriert.



(a) Der Ursprung des Fahrzeugkoordinatensystem liegt an einem definierten Punkt des Fahrzeugs

(b) Das Fahrzeugkoordinatensystem wird in dem Weltkoordinatensystem bewegt und rotiert

Abb. 2.1: Fahrzeug- und Weltkoordinatensystem in der Fahrzeugebene

Die Approximation des Fahrspurmodells wird im Fahrzeugkoordinatensystem durchgeführt. Eine anschließende Integration in das Weltkoordinatensystem kann zur Kartographierung durchgeführt werden. Die notwendige Transformation der Fahrspurmesswerte aus dem Kamerabild in das Fahrzeugkoordinatensystem wird in Abschnitt 3.3 beschrieben.

Der Ursprung des Fahrzeugkoordinatensystems wird in dieser Arbeit direkt mittig vor dem Fahrzeug platziert. Die vertikale Achse entspricht der Fahrzeuglängsachse (vgl. Abbildung 2.1a). Aus der Sicht des Fahrzeugs verläuft eine Kurve im Koordinatensystem vertikal. Damit die mathematische Abbildung der Kurve durch eine Funktion der Form: $f(x) = y$ dargestellt werden kann, wird die X-Achse senkrecht und die Y-Achse waagerecht eingezeichnet. Als Einheit des Fahrzeugkoordinatensystems wird Zentimeter festgelegt.

2.3 Fahrspurmodellierung durch ein Polynom 3. Grades

In zahlreichen Veröffentlichungen wird die quadratische Parabel

$$p(x) = \frac{1}{2} * C * x^2 + m * x + o \tag{2.1}$$

zur Fahrspurmodellierung in statischen Verfahren eingesetzt [Karl Kluge und Charles Thorpe (1992), Karl Kluge (1994), Karl Kluge und Sridhar Lakshmanan (1995), Claudio Rosito Jung und Christian Roberto Kelber (2004), Zhu Wennan u. a. (2006)]. Die Koeffizienten a, b und c in Gleichung (2.2) werden in dieser Arbeit als Parameter bezeichnet. Die quadratische Parabel basiert auf einer vereinfachten Reihenentwicklung zur Annäherung eines Halbkreises [Karl Kluge und Charles Thorpe (1992)]. Für die Bestimmung der Parabelparameter wird in [Karl Kluge und Charles Thorpe (1992)] und [Karl Kluge (1994)] die *Methode der kleinsten Quadrate* auf Teilmengen der Messwerte angewendet und aus den entstehenden Residuenvektoren (vgl. Abschnitt 3.4.1) der Median gebildet. Dadurch werden stark gestörte Messwerte gefiltert. In [Claudio Rosito Jung und Christian Roberto Kelber (2004)] und [Zhu Wennan u. a. (2006)] wird auf die Medianbildung verzichtet. Eine iterative, stochastische Parameterbestimmung durch Maximierung einer Wahrscheinlichkeitsfunktion wird alternativ in [Karl Kluge und Sridhar Lakshmanan (1995)] durchgeführt.

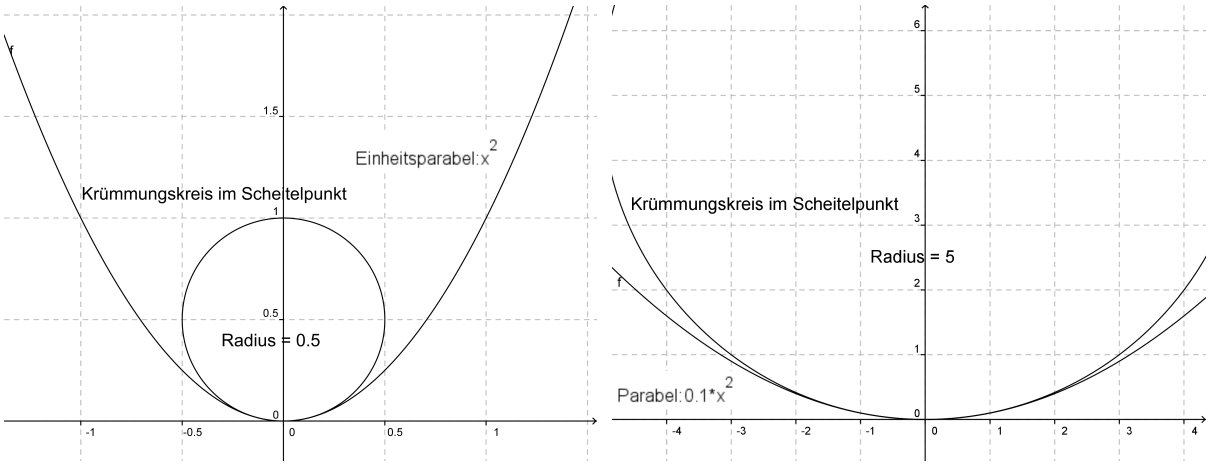
Die Krümmung C in Gleichung (2.1) bezieht sich auf den Krümmungskreis im Scheitelpunkt der Parabel (vgl. Abbildung 2.2). Der Radius des Krümmungskreises entspricht dem Kehrwert von C. Je geringer die Krümmung ist, umso mehr ist die Parabel gestaucht und umso größer ist der Krümmungskreis (vgl. Abbildung 2.2b). Die Parabel ist nur bei einer kleinen Krümmung nahezu Deckungsgleich mit dem Krümmungsreis.

Eine Fahrspur besteht nur selten aus einem Kreis. Hingegen stellt sie häufig eine Zusammensetzung aus verschiedenen Segmenten dar. Als Beispiel sei eine Fahrspur bestehend aus einer Geraden (Radius = ∞) gefolgt von einer Kurve (Radius = r) genannt. Die Anforderung einer Kreisannäherung durch die Parabel ist für eine Fahrspurapproximation aus diesem Grund nicht sinnvoll. Die Abkehr von der Kreisannäherung wird durch die nachfolgende Verwendung des allgemeineren Polynoms 2. Grades, oder auch quadratischen Polynoms (vgl. Gleichung 2.2) ausgedrückt:

$$p(x) = a * x^2 + bx + c \tag{2.2}$$

Das quadratische Polynom bietet nicht genügend Freiheitsgrade zur Approximation von S-Kurven, da es lediglich eine Krümmung in eine Richtung aufweist. Mathematisch begründet sich dies durch die konstante 2. Ableitung:

$$p''(x) = 2a \tag{2.3}$$



(a) Ungenaue Annäherung der Parabel im Betrachtungsbereich durch den Krümmungskreis bei kleinen Radien

(b) Relativ genaue Annäherung der Parabel im Betrachtungsbereich durch den Krümmungskreis bei größeren Radien

Abb. 2.2: Annäherung des Parabelradius durch den Krümmungskreis

Zudem zeigt die messtechnische Auswertung für das Beispiel der Geraden mit anschließender Kurve eine ungenaue Fahrspurapproximation durch das quadratische Polynom (vgl. Abschnitt 5.2.4), was in der Systemidentifikation zu erheblichen Messfehlern führt (vgl. Abschnitt 5.3).

In dieser Arbeit wird anstelle des quadratischen Polynoms ein kubisches Polynom mit einem weiteren Parameter

$$p(x) = a * x^3 + bx^2 + cx + d \tag{2.4}$$

verwendet. Der zusätzliche Parameter ermöglicht eine genauere Approximation des genannten Beispiels aus Gerade und Kurve (vgl. Abschnitt 5.2.4). Zudem kann das kubische Polynom gegenüber dem quadratischen Polynom S-Kurven approximieren, da die Richtung der Krümmung nicht konstant ist.

Durch die explizite Darstellung des Polynoms in Form einer Funktion von x können lediglich Kurven mit einem Richtungswinkel kleiner 90° dargestellt werden. Für die engste Kurve der Carolo-Cup Wettkampfstrecke [Carolo-Cup-Regelwerk (2009)] mit einem Radius von einem Meter entspricht das einem Bogen von maximal 1,57 Meter.

Das kubische Polynom stellt eine stetige Funktion mit ausreichend Freiheitsgraden zur Approximation von Kurven dar (vgl. Abschnitt 5.2.4). Zudem existieren numerische Standardverfahren zur Bestimmung der Parameter (vgl. Abschnitt 3.4).

2.4 Vorhersage der Polynomparameter für das Folgebild

Durch die Vorhersage der Polynomparameter wird die Lage und Orientierung der approximierten Fahrspurmarkierungen im Fahrzeugkoordinatensytem (vgl. Abschnitt 2.2) in Abhängigkeit der Zeit berechnet. Die gewonnenen Informationen werden zur Neuausrichtung der Betrachtungsbereiche im Kamerabild verwendet, um die Fahrspurmarkierungen verfolgen zu können (vgl. Abschnitt 3.1.2 und Abbildung 3.7).

Die Veränderung der Approximationspolynome lässt sich im Fahrzeugkoordinatensystem als eine affine Transformation bestehend aus einer Rotation und einer Translation (Verschiebung) ausdrücken. Ein Punkt $P = (x,y)$ auf dem Approximationspolynom wird in einen Punkt $P_t = (x_t, y_t)$ transformiert [Peter Hartmann (2006), Seite 197]:

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} \cos(\varphi) & -\sin(\varphi) \\ \sin(\varphi) & \cos(\varphi) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} \tag{2.5}$$

φ ist der Rotationswinkel. Ein $\varphi < 0$ wird nachfolgend als Rotation des Approximationspolynoms nach links und $\varphi > 0$ als Rotation nach rechts interpretiert. Die Rotation der Approximationspolynome im Fahrzeugkoordinatensystem erfolgt entgegengesetzt zur Rotation des Fahrzeugs. Da φ nachfolgend den Rotationswinkel des Fahrzeugs beschreibt, muss das Vorzeichen in der Transformation umgekehrt werden (vgl. Gleichung (2.6)). a und b beschreiben die Translation. Die Translation erfolgt ebenfalls entgegengesetzt zur Fahrzeugrichtung (vgl. Abbildung 2.3). Aus diesem Grund werden a und b subtrahiert:

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} \cos(-\varphi) & -\sin(-\varphi) \\ \sin(-\varphi) & \cos(-\varphi) \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} a \\ b \end{pmatrix} \tag{2.6}$$

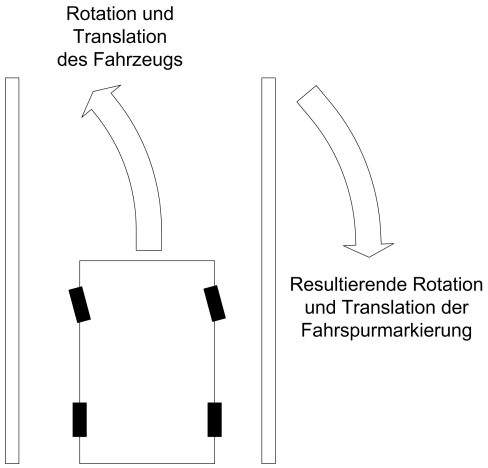


Abb. 2.3: Entgegengesetzte Rotation und Translation von Fahrzeug und Fahrspurmarkierungen

Die einfache Berechnung von φ durch die jeweilige Distanzmessung der Vorderräder des Fahrzeugs [Andrej Rull (2008), Abschnitt 4.4.1.2] ist nicht möglich. Der, gegenüber der Fahrspurerkennung, langsamere Zyklus der Distanzmesswerterzeugung (vgl. Abschnitt 5.1) bewirkt einen scheinbaren Fahrzeugstillstand für einige Fahrspurerkennungszyklen. Aus diesem Grund wird zur Bestimmung des Rotationswinkels φ, sowie der Translationskomponenten a und b ein dynamisches, kinematisches Einspurfahrzeugmodell eingesetzt [Denis Schetler (2007), Kapitel 5].

Die Werte für den Rotationswinkel und die Translation sind unbekannt und müssen aus den bekannten Systemvariablen

- υ, der Geschwindigkeit des Fahrzeugs in $\left[\frac{m}{s}\right]$, wobei $\upsilon \geq 0$ eine Vorwärtsfahrt und $\upsilon < 0$ einer Rückwärtsfahrt entspricht,

- α, der Lenkwinkel des Fahrzeugs in $[deg]$, wobei $\alpha < 0$ einem Lenkausschlag nach links und $\alpha > 0$ einem Lenkausschlag nach rechts entspricht,

- Δt, dem Zeitintervall zwischen zwei Bildauswertungen in $[s]$,
- L, dem Abstand des Vorderradachse zur Hinterradachse in $[m]$

berechnet werden. Die Systemvariablen υ, α, Δt und L werden nachfolgend als genau messbar und konstant zwischen dem aktuellen und dem Folgebild vorausgesetzt.

Als Sonderfall wird der Lenkwinkel $\alpha = 0$ betrachtet, wodurch sich der Rotationsteil der Transformation vereinfacht. Die Translation ergibt sich in diesem Fall direkt aus der Geschwindigkeit υ multipliziert mit dem Zeitintervall Δt

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} a \\ b \end{pmatrix} \quad \text{mit} \quad a = \upsilon * \Delta t \quad \text{und} \quad b = 0 \tag{2.7}$$

da das Fahrzeug keinen Bogen fährt. Bezogen auf das Fahrzeugkoordinatensystem ist a die Translation in x-Richtung und b die Translation in y-Richtung. Durch $b = 0$ fährt das Fahrzeug geradeaus. Die Darstellung des kubischen Polynoms (vgl. Abschnitt 2.3) als

$$\begin{aligned} p(x) &= p_1 * (x - x_t)^3 + p_2 * (x - x_t)^2 + p_3 * (x - x_t) + p_4 \\ &= p_1 * (x^3 - 3x^2 x_t + 3x x_t^2 - x_t^3) + p_2 * (x^2 - 2x x_t + x_t^2) + p_3 x_t - p_3 x_t + p_4 \\ &= p_1 * x^3 + (p_2 - 3p_1 x_t) * x^2 + (p_3 - 2p_2 x_t + 3p_1 x_t^2) * x + (p_4 - p_3 x_t + p_2 x_t^2 - p_1 x_t^3) \end{aligned} \tag{2.8}$$

erlaubt die unmittelbare Bestimmung der Polynomparameter für das Folgebild f:

$$\begin{aligned} p_{1_f} &= p_1 \\ p_{2_f} &= p_2 - 3p_1 x_t \\ p_{3_f} &= p_3 - 2p_2 x_t + 3p_1 x_t^2 \\ p_{4_f} &= p_4 - p_3 x_t + p_2 x_t^2 - p_1 x_t^3 \end{aligned} \tag{2.9}$$

Bei einem Lenkwinkel $\alpha \neq 0$ wird das kinematische Einspurfahrzeugmodell zur Berechnung von φ, a und b benötigt, da das Fahrzeug eine gekrümmte Strecke zurücklegt. Nachfolgend werden direkt die Lösungsgleichungen dargestellt. Die Herleitung dieser Gleichungen lässt sich in [Denis Schetler (2007), Kapitel 5] nachvollziehen.

Sind υ und α konstant im Zeitintervall und φ mit 0° initialisiert, so gilt für φ nach dem Zeitintervall Δt[Denis Schetler (2007), Gleichung (18)]:

$$\varphi = \frac{1}{L} * [\Delta t * \upsilon * \sin(\alpha)] \tag{2.10}$$

Für die Anfangsposition $P_a = (x_a, y_a)$ und die Endposition $P_e = (x_e, y_e)$ auf dem gefahrenen Bogen ergibt sich bei identischen Startbedingungen wie für Gleichung (2.10)

$$x_e = x_a + \frac{1}{2} * \Delta t * \upsilon * \cos(\alpha) + \frac{1}{2} * \Delta t * \upsilon * \cos(\varphi + \alpha), \tag{2.11}$$

sowie

$$y_e = y_a + \frac{1}{2} * \Delta t * \upsilon * \sin(\alpha) + \frac{1}{2} * \Delta t * \upsilon * \sin(\varphi + \alpha) \tag{2.12}$$

[Denis Schetler (2007), Gleichungen 19 und 20]. Aus den Differenzen zwischen x_a und x_e, sowie y_a und y_e lassen sich a und b für die Translation berechnen:

$$a = x_e - x_a \quad \text{und} \quad b = y_e - y_a \tag{2.13}$$

Der Startpunkt P_a wird mit $(0,0)$ initialisiert.

Durch eine Linearisierung der Sinus- und Kosinusfunktionen für kleine φ wird der Rechenaufwand der Transformation reduziert, sodass der transformierte Punkt P_t mit

$$\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} 1 & \varphi \\ -\varphi & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} a \\ b \end{pmatrix} \quad \text{mit} \quad \varphi = \varphi * \frac{\pi}{180} \tag{2.14}$$

berechnet wird.

Um den Rechenaufwand der Transformation zusätzlich zu verringern, werden nur einige Punkte eines Approximationspolynoms rotiert und verschoben. Anschließend wird mit den transformierten Punkten ein lineares Gleichungssystem aufgestellt und durch eine Ausgleichsrechnung die Parameter des kubischen Polynoms für das Folgebild approximiert (vgl. Abschnitt 3.4).

3 Approximation der Fahrspurmodellparameter

Bevor das Fahrspurmodell (vgl. Abschnitt 2.3) zur Identifikation (vgl. Kapitel 4) eingesetzt werden kann, müssen die unbekannten Parameter des Modells aus den Bilddaten approximiert werden. Dieses Kapitel beschreibt die einzelnen Verfahrensschritte zur Approximation in der Reihenfolge ihrer Ausführung.

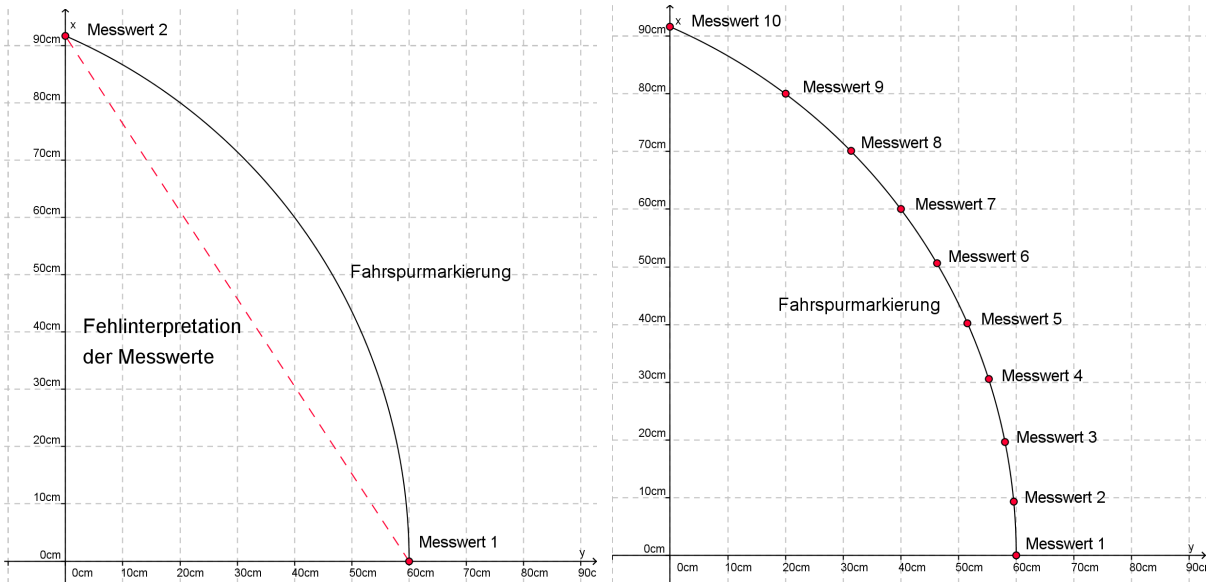
3.1 Erzeugung diskreter Messwerte durch Fahrspurerkennung im Kamerabild

Der erste Verfahrensschritt zur Approximation des Fahrspurmodells besteht aus der Erkennung der Fahrspurmarkierungen zur Erzeugung diskreter Messwerte für die weitere Verarbeitung. In Abschnitt 3.1.1 wird die bislang eingesetzte Fahrspurerkennung „TFALDA“ [Young Uk Yim und Se-Young Oh (2003)] vorgestellt und bewertet. Zur Bewertung werden die Genauigkeit der Diskretisierung, die Robustheit bezüglich gestörter Bilddaten und der Rechenaufwand betrachtet. Abschnitt 3.1.2 beschreibt die Vorzüge einer zeilenbasierten Fahrspurerkennung gegenüber „TFALDA“ bezogen auf die Bewertungskriterien. Im weiteren Verlauf dieser Arbeit findet das zeilenbasierte Verfahren Verwendung.

Die Diskretisierung der Fahrspurmarkierung erfolgt durch Einteilen eines Bildes in Betrachtungsbereiche. Ein Betrachtungsbereich ist ein rechteckiger Ausschnitt des Bildes. Er wird zur Reduktion der betrachteten Datenmenge eingesetzt. Im Englischen wird der Betrachtungsbereich als *Region Of Interest*(ROI) bezeichnet.

Um Veränderungen der Fahrspurmarkierungen im Bild verfolgen zu können, werden die Betrachtungsbereiche an der gefundenen Markierung neu ausgerichtet. Andernfalls könnte die Fahrspurmarkierung im nächsten Bild außerhalb eines Betrachtungsbereichs liegen und nicht mehr erkannt werden. Die beste Approximation der Fahrspurmarkierung innerhalb eines Betrachtungsbereichs wird nachfolgend als lokales Optimum bezeichnet. Analog wird die beste Approximation der Fahrspurmarkierung im gesamten Bild nachfolgend als globales Optimum bezeichnet.

Der Abstand der diskreten Fahrspurmarkierungswerte folgt aus der Dichte der Betrachtungsbereiche zueinander. Die Dichte legt die Abtastrate fest. Die Auswertung aller Bildzeilen ohne Verwenden von Betrachtungsbereichen entspricht der maximal möglichen Abtastung, da das Kamerabild bereits eine diskrete Darstellung der Fahrspurmarkierungen darstellt. Wird die Abtastrate geringer, besteht die Gefahr einer ungenauen Diskretisierung. Die Erkennung des Fahrspurverlaufs ist in diesem Fall nicht sicher gewährleistet (vgl. Abbildung 3.1a). Wird die Ab-



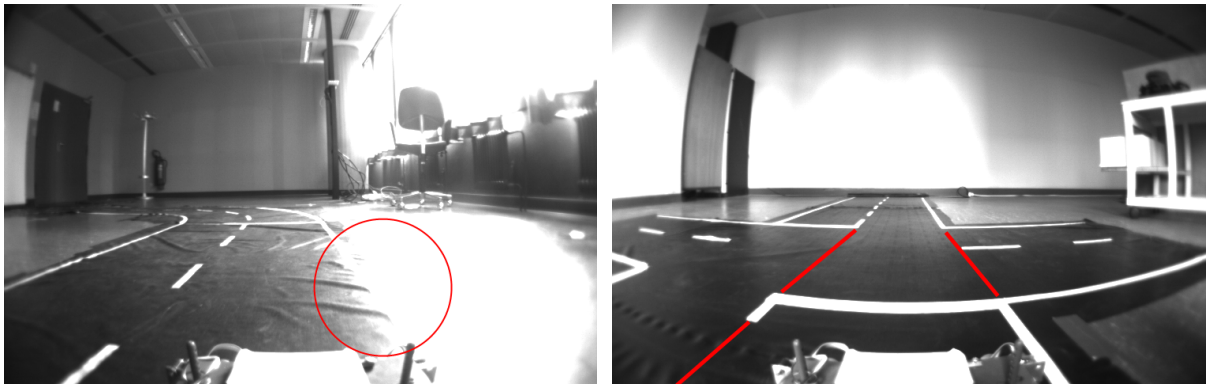
(a) Fehlerhafte Interpretation einer Kurve durch ungenaue Diskretisierung

(b) Korrekte Interpretation einer Kurve durch genaue Diskretisierung

Abb. 3.1: Einfluss der Abtastrate auf die Genauigkeit der Diskretisierung

tastrate höher, steigt die Wahrscheinlichkeit der korrekten Erkennung (vgl. Abbildung 3.1b), allerdings müssen zu Lasten der Verarbeitungsgeschwindigkeit mehr Daten verarbeitet werden.

Bei gestörten Bilddaten kann ein Fahrspurerkennungsverfahren trotz guter Abtastung ungenaue Ergebnisse liefern. Reflektionen verändern den Kontrast der Fahrspurmarkierung zur Fahrbahn und überdecken dadurch die Markierung (vgl. Abbildung 3.2a). Unterbrochene Fahrspurmarkierungen (vgl. Abbildung 3.2b) führen zu Betrachtungsbereichen ohne Markierungsinformationen. In beiden Fällen muss die Fahrspurerkennung das Fehlen einer Markierung kompensieren können.



(a) Kontrastveränderung der rechten Fahrspurmarkierung durch Reflektionen

(b) Nichtkontinuierliche Fahrbahnmarkierungen erschweren die Erkennung

Abb. 3.2: Störeinflüsse auf die Fahrbahnerkennung

3.1.1 Die bestehende TFALDA-Fahrspurerkennung

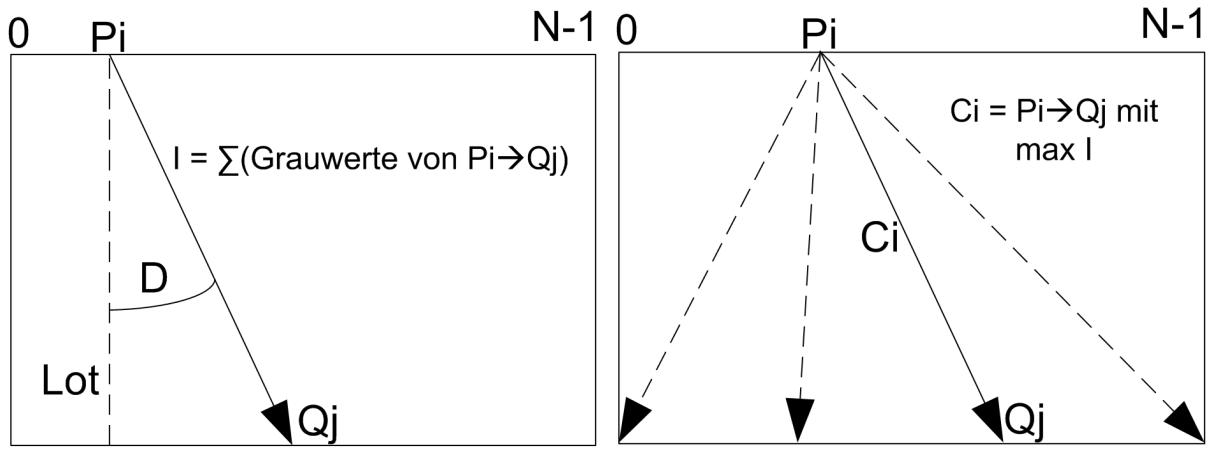
Im Rahmen der Teilnahme am Carolo-Cup [Carolo-Cup] im Februar 2008 wurde der „Three-Feature Based Automatic Lane Detection Algorithm (TFALDA)“ [Young Uk Yim und Se-Young Oh (2003)] zur Fahrspurerkennung implementiert [Dennis Berger (2008a)]. Nachfolgend wird das Verfahren einführend beschrieben und anschließend bewertet.

Einführung in TFALDA

Die Diskretisierung der Fahrspurmarkierungen erfolgt in TFALDA durch die symmetrische Aufteilung des Bildes in vier rechteckige Betrachtungsbereiche [Dennis Berger (2008b), Abbildung 2.6]. Innerhalb eines Betrachtungsbereiches wird die Fahrspurmarkierung durch eine gerade Strecke approximiert. Die Strecke wird mit drei Eigenschaften versehen:

- P, der Anfangspunkt der Strecke. Dieser Punkt liegt immer auf der oberen Begrenzung des Betrachtungsbereichs.
- D, die Richtung der Strecke. Sie wird angegeben durch den Winkel α zwischen der Strecke und dem Lot durch P.
- I, die Grauwertintensität der Strecke. I ist die Summe der Grauwerte entlang der Strecke.

In [Young Uk Yim und Se-Young Oh (2003)] wird die Strecke als Vektor bezeichnet. Abbildung 3.3a stellt die Eigenschaften grafisch dar.



(a) Eigenschaften eines Vektors im Betrachtungsbereich (b) Festlegen des Kandidaten C_i für ein P_i

Abb. 3.3: Vektoreigenschaften und Kandidaten in TFALDA

Vor der Bestimmung des Vektors wird der Betrachtungsbereich zur weiteren Verarbeitung vorbereitet [Dennis Berger (2008a), Abschnitte 2.1.2 und 2.1.3]. Zunächst reduziert eine Skalierung die Anzahl zu verarbeitender Pixel, anschließend wird der Sobeloperator (vgl. Abbildung 3.5) zur Kantenextraktion eingesetzt.

Die Bestimmung des Vektors im vorbereiteten Betrachtungsbereich der Breite N erfolgt in zwei Schritten:

1. **Kandidatensuche**: Von einem Punkt P_i mit $i = 0, \cdots, N-1$ werden die Strecken $\overline{P_iQ_j}$ zu allen Punkten Q_j mit $j = 0, \cdots, N-1$ gebildet (vgl. Abbildung 3.3b). Ein Punkt Q_j liegt immer auf der unteren Begrenzung des Betrachtungsbereichs. Der Kandidat C_i mit $P_{C_i} = P_i$ ist die Strecke $\overline{P_iQ_j}$ mit maximaler Grauwertintensität. Sobald der Kandidat für einen Punkt P_i gefunden wurde, wird der Kandidat für den Punkt P_{i+1} ermittelt. Auf diese Weise wird für alle P_i ein Kandidat bestimmt.

2. **Vektorselektion**: Alle Kandidaten C_i werden mit dem Vektor V aus dem selben Betrachtungsbereich im vorherigen Bild verglichen. Die Eigenschaften P, D und I werden für den Vergleich unterschiedlich gewichtet. Ein evolutionärer Algorithmus bestimmt die Gewichte K_P, K_D und K_I anhand von Testdaten [Young Uk Yim und Se-Young Oh (2003), Abschnitt 3]. Die Vergleichsrechnung erfolgt durch:

$$\lambda_{C_i} = K_P * |P_{C_i} - P_V| + K_D * |D_{C_i} - D_V| + K_I * (I_V - I_{C_i}) \tag{3.1}$$

   Der neue Vektor für den Betrachtungsbereich wird derjenige mit dem niedrigsten λ. Die Auswertung der I-Elemente erfolgt ohne Betragsstriche, um Vektoren mit hohem I zu bevorzugen. Ist beispielsweise $I_V < I_{C_i}$, so wird der letzte Term in (3.1) negativ und verringert die Größe von λ. Aus diesem Grund kann λ auch negative Werte annehmen.

In jedem Betrachtungsbereich im Bild wird durch die Kandidatensuche und die Vektorselektion ein optimaler Nachfolgevektor zum vorherigen Bild gesucht. Der Nachfolgevektor stellt das lokale Optimum dar.

Das „LaneInferenceSystem“ [Dennis Berger (2008b)] wertet die vier Vektoren aus den Betrachtungsbereichen aus und bildet die linke und rechte globale Fahrspurapproximation. Es dient als Ausgleich für Fehler bei der Bestimmung der Nachfolgevektoren, indem Bedingungen an den Fahrspurverlauf gestellt werden:

1. Die Fahrspurbreite ändert sich nicht sprunghaft.

2. Eine Fahrspurmarkierung verläuft stetig.

Die Fahrspurbreite wird jeweils horizontal zwischen den Endpunkten der linken und rechten Vektoren gemessen, sodass sich vier Messwerte ergeben [Dennis Berger (2008b), Abbildung 2.4]. Da die Vektoren direkt im Kamerabild liegen, nimmt die Fahrspurbreite zwischen den Endpunkten von oben nach unten aufgrund des Fluchtpunkts zu. Des Weiteren führt, vor allem in Fahrspurkurven, die horizontale Auswertung der Fahrspurbreite in der Bildfolge zu Differenzen in dem Abstand zwischen zwei Endpunkten [Dennis Berger (2008b), Abschnitt 2.1.1]. Die Differenz ist eine Konsequenz der Kameraperspektive. Sie entsteht auch, wenn die Fahrspur fehlerfrei approximiert wird.

Ein Tiefpassfilter glättet vor der weiteren Verarbeitung die vier gemessenen Fahrspurbreiten unter Berücksichtigung der Bildfolge [Dennis Berger (2008b), Abschnitt 2.1.2]. Die Parameter für den Filter ergeben sich aus den Spezifikationen der Carolo-Cup Wettkampfstrecke [Carolo-Cup-Regelwerk (2009)], der perspektivischen Abbildung durch die installierte Kamera sowie festgelegeten Anforderungen für die Fahrzeug- und Verarbeitungsgeschwindigkeit.

In einem Entscheidungssystem wird nach der Tiefpassfilterung die Fahrspur validiert. Sobald die Fahrspurbreite zwischen zwei Vektorendpunkten zu groß oder zu klein ist, werden die entsprechenden Vektoren durch ihre bereits validierten Vorgänger aus dem vorherigen Bild ersetzt. Ebenso wird verfahren, wenn die Differenz der Fahrspurbreite vom vorherigen zum aktuellen Bild zu groß ist. Die Grenzwerte für die Validierung wurden empirisch festgelegt.

Zur Nachbildung der Fahrspurstetigkeit mit den gültigen Vektoren wird eine vertikale Korrektur durchgeführt [Dennis Berger (2008b), Abschnitt 2.2]. Der Startpunkt P_i (vgl. Abbildung 3.3) des unteren Vektors wird zu Korrektur am Endpunkt Q_j des oberen Vektors ausgerichtet. Der obere Vektor wird priorisiert, da sich am Eingang in einer Fahrspurkurve dessen Richtung zuerst verändert.

Nachdem das „LaneInferenceSystem“ die Fahrspurapproximation validiert und gegebenenfalls korrigiert hat, werden die Betrachtungsbereiche für das nachfolgende Bild positioniert [Dennis Berger (2008b), Abschnitt 2.3]. Die Endpunkte der Vektoren werden zur weiteren Verarbeitung in der Regelungssoftware gespeichert.

Ausschlussgründe für TFALDA

Bei der Approximation einer kurvigen Fahrspurmarkierung durch Geraden, ist der Fehler in der Diskretisierung von der Länge der Approximationsgeraden abhängig. Die in [Young Uk Yim und Se-Young Oh (2003)] und [Dennis Berger (2008b)] gewählte Bildaufteilung in vier Betrachtungsbereiche erweist sich als nachteilige Diskretisierung (vgl. Abbildung 3.4). Für eine weite Sicht auf die Fahrspur müssen die Betrachtungsbereiche entsprechend hoch sein, wodurch enge Kurven ungenau approximiert werden. Eine Polynomapproximation durch die Vektorendpunkte aus Abbildung 3.4 würde bereits ohne Erkennungsfehler zu einer falschen Lösung führen.

Der Approximationsfehler durch die Vektoren lässt sich mit einer besseren Diskretisierung verringern. Eine Aufteilung des Bildes in beispielsweise acht Betrachtungsbereiche [Eike Jenning (2008a)] erfordert die nachträgliche Anpassung des „LaneInferenceSystem“ und der Gewichte für die Vektorselektion.

Für das „LaneInferenceSystem“ müssen die Fahrspurbreitenvalidierung, die vertikale Ausrichtung und die Neupositionierung der Betrachtungsbereiche überarbeitet werden.

Die Gewichte für die Vektorselektion werden unter vorgegebener Bildaufteilung von einem evolutionären Algorithmus anhand eines Trainingsdatensatzes bestimmt. Zur Laufzeit von TFALDA müssen aus diesem Grund zum Training vergleichbare Bedingungen vorherrschen, andernfalls sind die Gewichte nicht optimal. Eine Veränderung der Bildaufteilung erfordert demnach auch eine Neuberechnung der Gewichte, zumal die bislang verwendeten Gewichte aus [Young Uk Yim und Se-Young Oh (2003)] aus einem größtenteils unbekannten Trainingsdatensatz stammen.

Die genannten Anpassungen verbessern TFALDA allerdings nicht konzeptionell. Vielmehr wird die konzeptionelle Schwäche besser kompensiert.

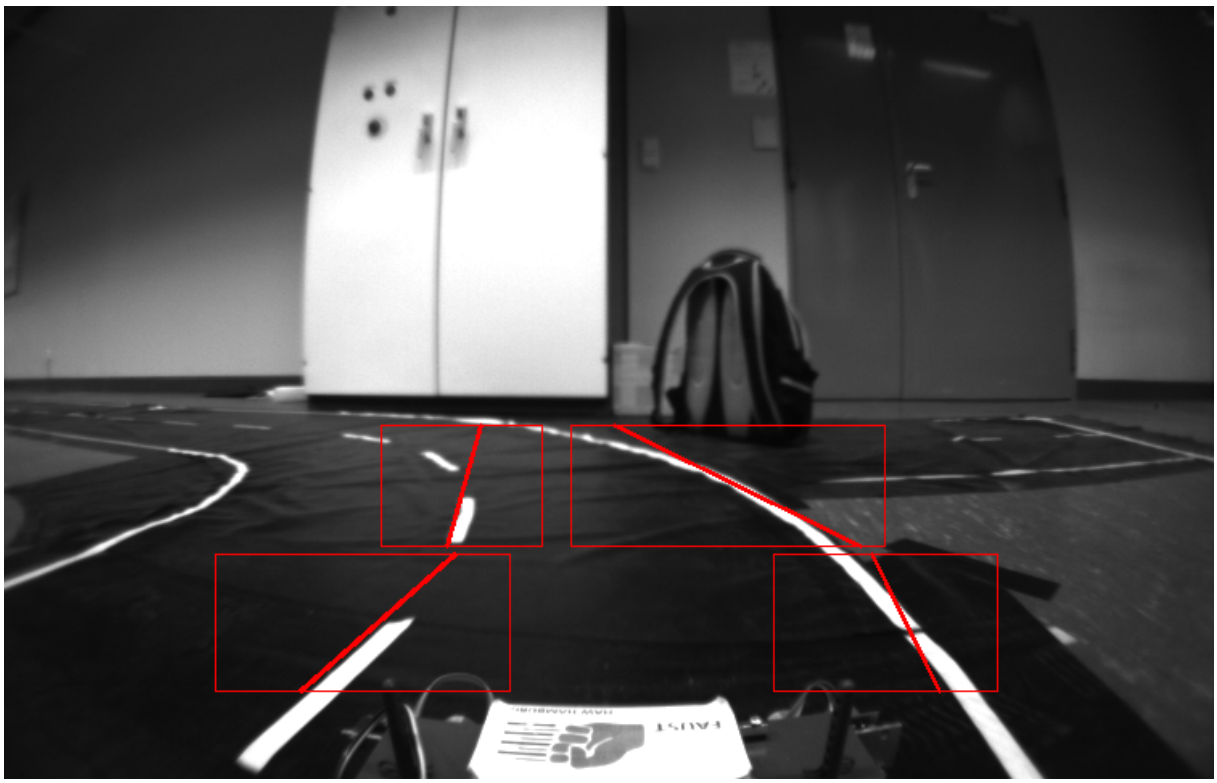


Abb. 3.4: Fehlerhaft Approximation durch schlechte Diskretisierung

Konzeptionell vereint TFALDA eine lokale und eine globale Optimierung von Vektoren zur Approximation der Fahrspur. Die globale Optimierung besteht aus der Validierung und Korrektur der lokalen Optima. Dieses Konzept führt sowohl zu einem hohen Rechenaufwand (vgl. Abschnitt 5.4), als auch zu einem Widerspruch zwischen den Optima.

Die globale Optimierung durch das „LaneInferenceSystem“ ist unerlässlich, da erst in dieser Verarbeitungsstufe aus Geradenstücken kontrolliert, unter Berücksichtigung von Fahrspureigenschaften(zum Beispiel Stetigkeit), die gesamte Fahrspurmarkierung approximiert wird.

Die Notwendigkeit der lokalen Optimierung durch Vektorselektion ist auf das Verfahren zur Kandidatensuche zurückzuführen. Da in jedem Fall die Menge der Kandidaten für die Begrenzungspunkte P_i eines Betrachtungsbereichs (vgl. Abbildung 3.3a) gebildet wird, muss aus dieser Menge auch der optimale Kandidat ausgesucht werden. Die Menge wiederrum ist erforderlich, weil das Verfahren keine Kenntnis über die Lage und Richtung der Fahrspurmarkierung im Betrachtungsbereich besitzt und somit nicht sofort auf den richtigen Vektor schließen kann.

Die Verwendung einzeiliger Betrachtungsbereiche (vgl. Abschnitt 3.1.2) reduziert eine gerade Strecke auf einen Punkt, sodass die Richtung nicht länger betrachtet werden muss. Die Menge der Kandidaten setzt sich dann lediglich aus den hellen Punkten des Betrachtungsbereichs zusammen. Wird die Breite des Betrachtungsbereichs etwas größer gewählt als die Fahrspurbreite kann die Auswahl eines Punktes vereinfacht werden, indem ein Fehler zugelassen wird. Die entstehende Varianz der Messwerte ist nicht größer als die Breite des Betrachtungsbereichs und wird in diesem Fall durch die globale Optimierung ausgeglichen.

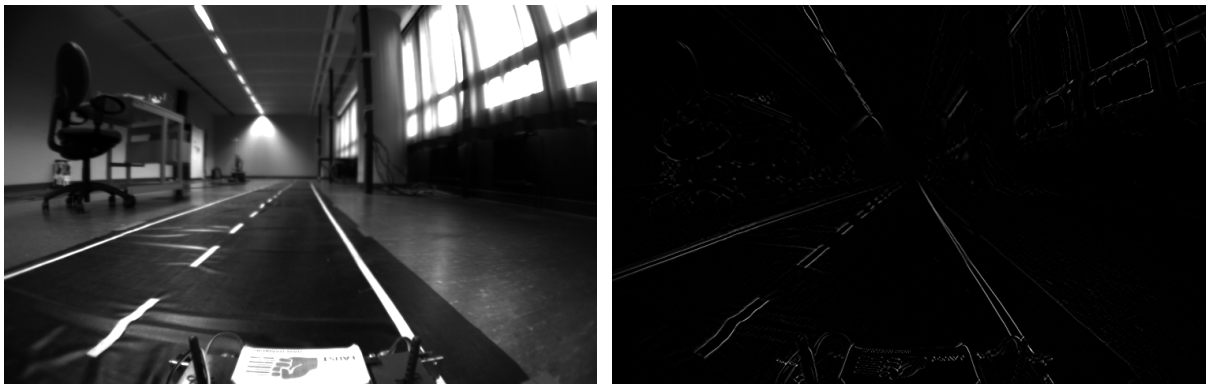
Statt die konzeptionelle Schwäche von TFALDA durch Anpassung der Diskretisierung und der Gewichtung auszugleichen, wird ein alternatives Fahrspurerkennungverfahren verwendet. In einzeiligen Betrachtungsbereichen wird die Fahrspurerkennung ohne lokale Optimierung

durchgeführt (vgl. Abschnitt 3.1.2). Die Ergebnisse werden anschließend optimal durch ein Polynom approximiert (vgl. Kapitel 3).

3.1.2 Einzeilige Betrachtungsbereiche zur Fahrspurerkennung

Anstelle von TFALDA (vgl. Abschnitt 3.1.1) wird die Fahrspurmarkierung durch einzeilige Betrachtungsbereiche diskretisiert [R. Risack u. a. (1998), Keith A. Redmill u. a. (2001)]. Zugunsten des Rechenaufwands wird auf eine lokale Optimierung verzichtet. Stattdessen wird die Anzahl der Betrachtungsbereiche pro Fahrspurmarkierung deutlich höher gewählt als die Anzahl der Parameter des Ausgleichspolynoms (vgl. Abschnitt 2.3), sodass ein überbestimmtes Gleichungssystem gelöst werden muss (vgl. Abschnitt 3.4). Ziel ist die Kompensation der Messwertvarianz aufgrund fehlender lokaler Optimierung durch das überbestimmte Gleichungssystem.

Innerhalb eines einzeiligen Betrachtungsbereichs wird das Kamerabild wie in TFALDA mit dem Sobeloperator (vgl. Abbildung 3.5) gefiltert und anschließend ausgewertet. Bedingt durch die Höhe des Sobeloperators von drei Pixeln werden die Bildzeilen über und unter dem Betrachtungsbereich ebenfalls zur Filterung herangezogen.



(a) Bild der Carolo-Cup Teststrecke (b) Betragsbild des Sobeloperators

Abb. 3.5: Sobeloperator zur Kantenextraktion

Ausgehend vom Mittelpunkt des Betrachtungsbereichs wird simultan in Richtung des linken und rechten Betrachtungsbereichrands durch die Pixel iteriert und das erste helle Pixel als Fahrspur selektiert (vgl. Abbildung 3.6). Sollten sowohl das linke als auch das rechte Pixel hell sein, entscheidet die Lage des Betrachtungsbereichs über die Auswahl. Der Betrachtungsbereich über der linken Fahrspurmarkierung wählt das rechte Pixel aus, der Betrachtungsbereich über der rechten Fahrspurmarkierung wählt das linke Pixel aus.

Die Auswertung wird vom Mittelpunkt aus gestartet, da der Betrachtungsbereich anhand der Approximation für das aktuelle Bild auf das nachfolgende Bild ausgerichtet wird. Die Ausrichtung erfolgt so, dass der Mittelpunkt genau auf dem vorhergesagtem Polynom für das Folgebild (vgl. Abschnitt 2.4) liegt. Die Fahrspur wird aus diesem Grund in der Nähe der Betrachtungsbereichsmitte erwartet.

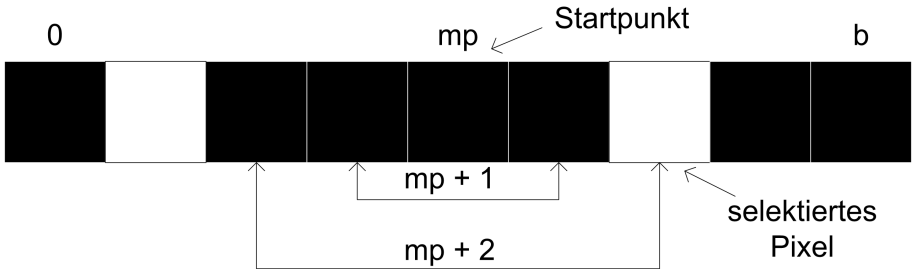


Abb. 3.6: Auswertung der Pixel eines Betrachtungsbereichs vom Mittelpunkt *mp* ausgehend

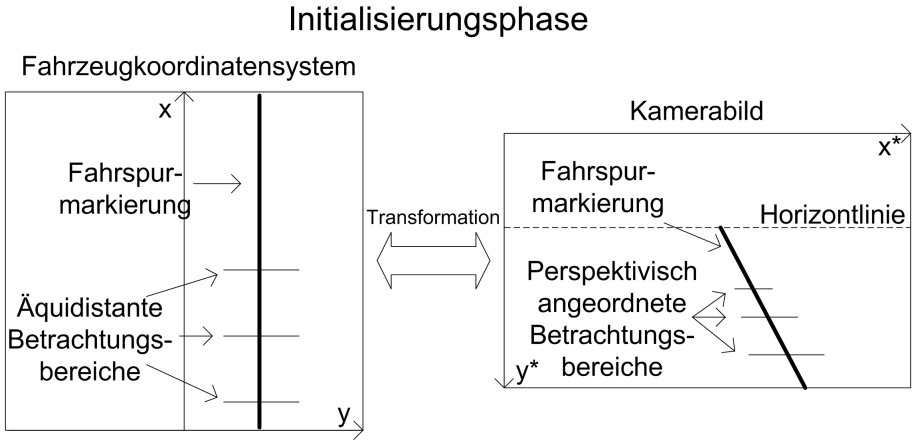
Die Anordnung der Betrachtungsbereiche im Fahrzeugkoordinatensystem (vgl. Abbildung 3.7a) erfordert eine Transformation der Positionen in das unkorrigierte Kamerabild (vgl. Abschnitt 3.2). Die in dieser Arbeit beschriebene Verfahrenskette wird dazu rückwärts durchlaufen (vgl. Abbildung 1.1). Während der Initialisierungsphase des Verfahrens werden mit der Ebenentransformation (vgl. Abschnitt 3.3) die Endpunkte der Betrachtungsbereiche aus dem Fahrzeugkoordinatensystem in das korrigierte Bild projeziert und daraus die Breite berechnet. Da der Betrachtungsbereich im korrigierten Kamerabild nicht schmaler als im unkorrigierten Kamerabild sein kann (vgl. Abbildung 3.8 für eine Veranschaulichung), wird die berechnete Breite für das unkorrigierte Bild übernommen. Zusätzlich wird der Mittelpunkt jedes Betrachtungsbereichs durch Ebenentransformation und Linsenverzeichnungskorrektur aus dem Fahrzeugkoordinatensystem in das unkorrigierte Bild transformiert und anschließend der Betrachtungsbereich mit der berechneten Breite aufgespannt. Nach der Initialisierungsphase wird zur Verminderung des Rechenaufwands lediglich die horizontale Verschiebung des Betrachtungsbereichmittelpunkts im Fahrzeugkoordinatensystem berücksichtigt. Der y-Wert des neuen Mittelpunkts wird in das unkorrigierte Bild transformiert. Die Breite und die vertikale Lage des Betrachtungsbereichs werden nicht verändert.

Der Betrachtungsbereich darf nach seiner Neuausrichtung nicht außerhalb des Bildes liegen (vgl. 3.7b). Seien n die Breite und mp der Mittelpunkt des Betrachtungsbereichs, sowie b die Breite des Bildes und x^* die horizontale Achse, so muss gelten:

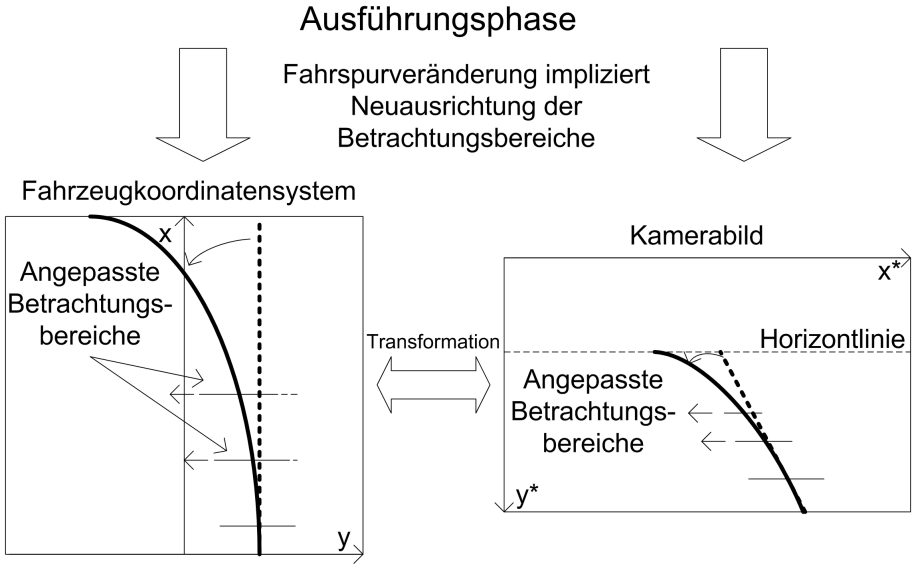
$$x^*_{mp} - \frac{n}{2} \geq 0 \quad \text{und} \quad x^*_{mp} + \frac{n}{2} \leq b \tag{3.2}$$

Ist eine der Bedingungen aus Gleichung 3.2 verletzt wird der Betrachtungsbereich an der Bildgrenze platziert und deaktiviert. Die anschließende Bildauswertung mit einer reduzierten Betrachtungsbereichsmenge muss durch die Überbestimmtheit so kompensiert werden, dass in engen Kurven die nahen Betrachtungsbereiche genügend Informationen zur korrekten Approximation erzeugen. Sobald beide Bedingungen aus Gleichung 3.2 erfüllt sind wird der Betrachtungsbereich wieder aktiviert.

Zwischen den Betrachtungsbereichen wird ein äquidistanter Abstand zur Diskretisierung der Fahrspur festgelegt. Aufgrund der Perspektive nimmt im Kamerabild die Dichte der Betrachtungsbereiche deshalb zum Horizont hin zu. Der maximale Abstand ohne fehlerhafte Diskretisierung ergibt sich aus der Betrachtung der Leitlinie der Carolo-Cup Wettkampfstrecke [Carolo-Cup-Regelwerk (2009)]. In einem Intervall von 20 Zentimetern folgen die Fahrspurmarkierung und freie Bereiche aufeinander.



(a) Aufbau der Betrachtungsbereiche in Fahrzeugkoordinatensystem und Kamerabild. Die Betrachtungsbereichsmittelpunkte werden in das linsenverzeichnete Kamerabild transformiert. Durch eine einmalige Berechnung der Betrachtungsbereichsbreite wird vom transformierten Mittelpunkt ausgehend der Betrachtungsbereich im Bild aufgespannt.



(b) Horizontale Neuausrichtung der Betrachtungsbereiche aufgrund einer Fahrspurveränderung. Zur Laufzeit wird lediglich der y-Wert des Betrachtungsbereichmittelpunkts aktualisiert und transformiert. Alle anderen Information sind bereits bekannt.

Abb. 3.7: Aufbau und Neuausrichtung der Betrachtungsbereiche

Für eine fehlerfreie Diskretisierung wird der Abstand zwischen zwei Betrachtungsbereichen auf 15 Zentimeter festgelegt. Um eine Sichtweite von etwa 2 Metern auf einer geraden Fahrspur zu erzielen werden 13 Betrachtungsbereiche verwendet. Eine erfolgreiche Approximation ist möglich, sobald die Anzahl der Betrachtungsbereiche größer oder gleich der Anzahl der Parameter des Ausgleichspolynoms (vgl. Abschnitt 2.3) ist. Der minimal notwendige Anteil P an fehlerfreien Messwerten ist bei den gegebenen Werten

$$P = \frac{3}{13} \approx 23\% \tag{3.3}$$

Bei einem geringen P müssen die Messwerte günstig verteilt sein, um Diskretisierungsfehler zu vermeiden (vgl. Abbildung 3.1a).

3.2 Linsenverzeichnungskorrektur der Messwerte zur Fehlerminimierung

Bedingt durch die Linsenkrümmung eines Kameraobjektivs entsteht im Bild eine Linsenverzeichnung [Nico Manske (2008), Abschnitt 4.2]. Ausgehend vom Bildhauptpunkt nimmt die Verzeichnung nach außen hin zu. Je weitwinkliger das Objektiv ist, umso stärker ist die Verzeichnung. Die Konsequenz ist eine fehlerhafte Darstellung der Realität. Abbildung 3.8a zeigt die Aufnahme einer Kamera mit weitwinkligem Objektiv. Gut zu erkennen ist die kreisförmige Darstellung einer Geraden im unteren Bildbereich.

Die Abbildung der Fahrspurmarkierungen erfolgt in dem verzeichnungsfreihen Fahrzeugkoordinatensystem (vgl. Abschnitt 2.2). Vor der Projektion der Fahrspurmesswerte in das Koordinatensystem (vgl. Abschnitt 3.3) wird aus diesem Grund eine Korrektur der Linsenverzeichnung durchgeführt.

Durch eine einmalige Kalibrierung der Kamera werden die, zur Verzeichnungskorrektur benötigten, internen Kameraparameter bestimmt. Aufgrund des Herstellungsprozesses variieren die internen Kameraparameter selbst bei baugleichen Kameras. Die Berechnung der Parameter erfolgt durch eine photogrammetische Vermessungssoftware mit Hilfe eines ausgedruckten Kalibriermusters [Nico Manske und Thorsten Jost (2008), Abschnitt 3]. Die Kalibrierung der für den Prototyptest eingesetzten Kamera wird im Abschnitt 5.1 beschrieben.

Die Linsenverzeichnungskorrektur wird sowohl zur Transformation von verzeichneten in verzeichnungsfreie Bildpunkte, als auch zur Rücktransformation verzeichnungsfreier in verzeichnete Bildpunkte benötigt (vgl. Abbildung 1.1 und Abschnitt 3.1.2). Nachfolgend wird zunächst die Transformation in verzeichnungsfreie Punkte beschrieben. Anschließend werden die Gleichungen zur Rücktransformation umgestellt und gelöst.

Für die Berechnung der verzeichnungsfreien Bildpunkte werden folgende interne Kameraparameter benötigt:

- die Bildhauptpunktkoordinaten $P_{bh} = (x_{bh}, y_{bh})$ in Millimetern,
- und die Linsenverzeichnungsparameter K_1, K_2, P_1, P_2.

Die Verzeichnungskorrektur lässt sich in drei Berechnungsschritte gliedern [entnommen aus Nico Manske (2008), Abschnitt 4.2]. Im ersten Schritt erfolgt die Transformation der Pixelkoordinaten eines verzeichneten Bildpunktes $P_b = (x_b, y_b)$ in einen Punkt $P_m = (x_m, y_m)$ im metrischen Koordinatensystem um den Bildhauptpunkt:

$$x_m = \frac{x_b}{Pix_w} - x_{bh} \tag{3.4}$$

$$y_m = \frac{y_b}{Pix_h} - y_{bh} \tag{3.5}$$

Pix_h ist die Anzahl an Pixeln pro Millimeter Chiphöhe. Pix_w ist die Anzahl an Pixeln pro Millimeter Chipbreite.

Die verzeichneten, metrischen Punktkoordinaten werden im zweiten Schritt in korrigierte metrische Punktkoordinaten $P_c = (x_c, y_c)$ umgerechnet:

$$x_c = x_m \cdot (1 + K_1 \cdot r^2 + K_2 \cdot r^4) + P_1 \cdot (r^2 + 2 \cdot x_m^2) + 2 \cdot P_2 \cdot x_m \cdot y_m \tag{3.6}$$

$$y_c = y_m \cdot (1 + K_1 \cdot r^2 + K_2 \cdot r^4) + P_2 \cdot (r^2 + 2 \cdot y_m^2) + 2 \cdot P_1 \cdot x_m \cdot y_m \tag{3.7}$$

mit

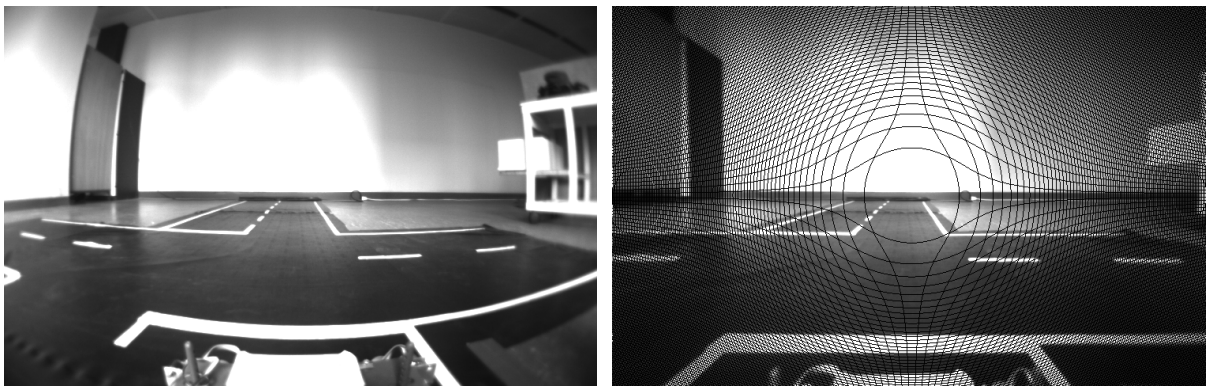
$$r^2 = x_m^2 + y_m^2 \tag{3.8}$$

Zuletzt müssen die korrigierten, metrischen Punktkoordinaten in einen Bildpunkt $P_{cb} = (x_{cb}, y_{cb})$ in Pixelkoordinaten rücktransformiert werden:

$$x_{cb} = (x_c + x_{bh}) \cdot Pix_w \tag{3.9}$$

$$y_{cb} = (y_c + y_{bh}) \cdot Pix_h \tag{3.10}$$

Der verzeichnungsfreihe Punkt P_{cb} wird gespeichert und zur weiteren Verarbeitung verwendet. Abbildung 3.8b zeigt das Resultat der Linsenverzeichnungskorrektur. Das schwarze Linienmuster entsteht dadurch, dass kein korrigierter Quellbildpunkt auf einer der Linien liegende Zielbildkoordinaten besitzt. Per Voreinstellung bleiben die Pixel im Zielbild an diesen Stellen schwarz.



(a) Linsenverzeichnung durch ein weitwinkliges Objektiv (b) Resultat der Linsenverzeichnungskorrektur

Abb. 3.8: Linsenverzeichnetes Quellbild und korrespondierendes, verzeichnungsfreies Zielbild

Für die Rückwärtstransformation verzeichnungsfreier Pixelkoordinaten in verzeichnete Pixelkoordinaten ist die Lösung eines nicht-linearen Gleichungssystems notwendig. Nachfolgend wird das Newton-Verfahren zur Näherung der richtigen Lösung verwendet. Die Lösung ist entnommen aus [Nico Manske (2008)].

Zunächst müssen die verzeichnungsfreien Pixelkoordinaten x_{cb} und y_{cb} wieder in das metrische Koordinatensystem um den Bildhauptpunkt transformiert werden:

$$x_c = \frac{x_{cb}}{Pix_w} - x_{bh} \tag{3.11}$$

$$y_c = \frac{y_{cb}}{Pix_h} - y_{bh} \tag{3.12}$$

Die verzeichnungsfreien, metrischen Punktkoordinaten werden anschließend in den umgestellten Korrekturfunktionen verwendet. Zur einfacheren Handhabung werden die Funktionen benannt:

$$f_1(x_m, y_m) = x_m \cdot (1 + K_1 \cdot r^2 + K_2 \cdot r^4) + P_1 \cdot (r^2 + 2 \cdot x_m^2) + 2 \cdot P_2 \cdot x_m \cdot y_m - x_c \tag{3.13}$$

$$f_2(x_m, y_m) = y_m \cdot (1 + K_1 \cdot r^2 + K_2 \cdot r^4) + P_2 \cdot (r^2 + 2 \cdot y_m^2) + 2 \cdot P_1 \cdot x_m \cdot y_m - y_c \tag{3.14}$$

Für die zwei Unbekannten x_m und y_m werden zur Lösung zwei Gleichungen benötigt. Mit den Funktionen f_1 und f_2 wird das Newton-Verfahren für Gleichungssysteme [Norbert Herrmann (2004), Abschnitt 6.6.3]

$$\begin{pmatrix} \frac{\partial f_1}{\partial x_m} & \frac{\partial f_1}{\partial y_m} \\ \frac{\partial f_2}{\partial x_m} & \frac{\partial f_2}{\partial y_m} \end{pmatrix} \cdot \begin{pmatrix} x_{mn+1} - x_{mn} \\ y_{mn+1} - y_{mn} \end{pmatrix} = \begin{pmatrix} -f_1(x_{mn}, y_{mn}) \\ -f_2(x_{mn}, y_{mn}) \end{pmatrix} \tag{3.15}$$

durchgeführt. Die partiellen Ableitungen von $f_1(x_m, y_m)$ und $f_2(x_m, y_m)$ sind zur besseren Übersicht in Anhang E aufgelistet. n kennzeichnet den aktuellen und $n+1$ den nächst genaueren Wert der Näherung. Das Verfahren wird iterativ ausgeführt und abgebrochen wenn

$$x_{mn+1} - x_{mn} \leq \varepsilon_x \quad \text{und} \quad y_{mn+1} - y_{mn} \leq \varepsilon_y \tag{3.16}$$

ist. Die Grenzwerte ε_x und ε_y für eine pixelgenaue Näherung berechnen sich aus den Parametern Pix_w und Pix_h:

$$\varepsilon_x = \frac{1}{Pix_w} \quad \text{und} \quad \varepsilon_y = \frac{1}{Pix_h} \tag{3.17}$$

Die Kehrwerte von Pix_w und Pix_h entsprechen den Millimetern pro Pixel im metrischen Bildhauptpunktkoordinatensystem. Für den ersten Iterationsschritt $n = 0$ müssen für x_m und y_m Startwerte vorgegeben werden. In diesem Fall werden als Startwerte die verzeichnungsfreien metrischen Koordinaten x_c und y_c eingesetzt:

$$x_{m0} = x_c \quad \text{und} \quad y_{m0} = y_c \tag{3.18}$$

Sobald das Newton-Verfahren die genäherten, verzeichneten, metrischen Koordinaten x_m und y_m bestimmt hat, werden diese in Pixelkoordinaten umgerechnet:

$$x_b = (x_m + x_{bh}) \cdot Pix_w \tag{3.19}$$

$$y_b = (y_m + y_{bh}) \cdot Pix_h \tag{3.20}$$

Gleichung (3.15) beschreibt ein lineares Gleichungssystem der Form

$$\mathbf{A} \cdot \vec{x} = \vec{b}, \tag{3.21}$$

wobei **A** als Jakobi-Matrix oder Funktionalmatrix bezeichnet wird. Zur Lösung des Gleichungssystems wird die Singulärwertzerlegung eingesetzt (vgl. Abschnitt 3.4.2).

Durch die gute Konvergenz des Newton-Verfahrens sind bei günstig liegenden Startwerten die Näherungen bei einer festen Anzahl von Iterationen in der Regel ausreichend genau. Zudem sollte zur weiteren Reduzierung des Rechenaufwands eine Speicherung der korrespondierenden, verzeichneten und verzeichnungsfreien Bildpunkte in Betracht gezogen werden.

3.3 Projektive Transformation zwischen Bild- und Fahrzeugkoordinatensystem

Zur Identifikation der Fahrzeug- und Fahrbahnorientierung (vgl. Kapitel 4) wird die Approximation der Fahrspur in dem Fahrzeugkoordinatensystem (vgl. Abschnitt 2.2) durchgeführt. Das Kamerabild der Fahrspur befindet sich auf einer anderen zweidimensionalen Ebene im Raum als das Fahrzeugkoordinatensystem (vgl. Abbildung 3.9). Aus diesem Grund werden die verzeichnungsfreien Messwerte (vgl. Abschnitt 3.2) vor der Approximation in das Fahrzeugkoordinatensystem transformiert.

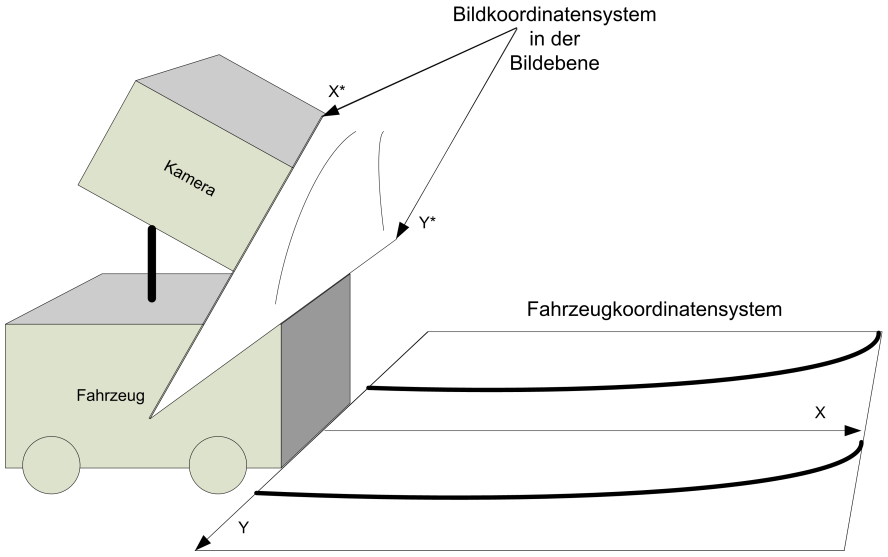


Abb. 3.9: Bild- und Fahrzeugkoordinatensystem im Raum

3.3.1 Projektive Transformation mit homogenen Koordinaten

Ein Punkt $\vec{p}_k$ in Vektorschreibweise mit kartesischen Koordinaten $(x_k \quad y_k)^T$ wird durch die Erweiterung um eine zusätzliche Dimension zu einem Punkt $\vec{p}_h$ in homogenen Koordinaten $(x_h \quad y_h \quad 1)^T$. Ein Koordinatenvektor ist homogen, wenn folgende Äquivalenz gilt:

$$\begin{pmatrix} x_h \\ y_h \\ 1 \end{pmatrix} \Leftrightarrow \begin{pmatrix} sx_h \\ sy_h \\ s \end{pmatrix}, \text{ für alle } s \neq 0 \tag{3.22}$$

s wird als Skalierungsfaktor des homogenen Koordinatenvektors bezeichnet. Zur Transformation von $\vec{p}_h$ zu $\vec{p}$ werden die homogenen Koordinaten durch den Skalierungsfaktor geteilt:

$$\vec{p}_h = \begin{pmatrix} sx_h \\ sy_h \\ s \end{pmatrix} \quad \rightarrow \quad \vec{p}_k = \begin{pmatrix} x_k = \frac{sx_h}{s} \\ y_k = \frac{sy_h}{s} \end{pmatrix} \tag{3.23}$$

Projektive Transformationen, oder auch Projektionen, beschreiben alle geometrischen Transformationen wie Rotation, Skalierung und Translation. Eine projektive Transformation beinhaltet isometrische, Ähnlichkeits- und affine Transformationen [Richard Hartley und Andrew Zisserman (2003), Abschnitt 2.4] und erweitert diese um die Unterstützung für Fluchtpunkte [Richard Hartley und Andrew Zisserman (2003), Abschnitt 2.4.5]. Die Berechnung der Projektion einer Ebene auf eine andere lässt sich mit homogenen Koordinaten als lineare Transformation der Form

$$\vec{x_h} = \mathbf{H}\vec{x_h^*} \quad \Leftrightarrow \quad \begin{pmatrix} x_h \\ y_h \\ 1 \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \cdot \begin{pmatrix} x_h^* \\ y_h^* \\ 1 \end{pmatrix} \tag{3.24}$$

darstellen. Die reguläre, homogene 3x3 Matrix $\mathbf{H}$ wird als Transformationsmatrix bezeichnet. Die Division aller Matrixelemente durch den Skalierungsfaktor h_{33} reduziert die Transformationsmatrix auf acht unbekannte Elemente. Sei $b_{ij} = (h_{ij} \,/\, h_{33})$

$$\vec{x_h} = \mathbf{B}\vec{x_h^*} \quad \Leftrightarrow \quad \begin{pmatrix} x_h \\ y_h \\ 1 \end{pmatrix} = \begin{pmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & 1 \end{pmatrix} \cdot \begin{pmatrix} x_h^* \\ y_h^* \\ 1 \end{pmatrix} \tag{3.25}$$

Die Berechnung der kartesischen Koordinaten $(x_k \quad y_k)^T$ ergibt sich aus Gleichung 3.23. Die Skalierung der homogenen Koordinaten sei jeweils 1. Es gilt:

$$x_k = \frac{x_h}{1} = \frac{b_{11}x_h^* + b_{12}y_h^* + b_{13}}{b_{31}x_h^* + b_{32}y_h^* + 1} \tag{3.26}$$

$$y_k = \frac{y_h}{1} = \frac{b_{21}x_h^* + b_{22}y_h^* + b_{23}}{b_{31}x_h^* + b_{32}y_h^* + 1} \tag{3.27}$$

Gemäß der Notation in Abbildung 3.9 befindet sich $\vec{x} = (x_k \quad y_k)^T$ im Fahrzeugkoordinatensystem und $\vec{x^*} = (x_k^* \quad y_k^*)^T$ im Bildkoordinatensystem. Somit lässt sich für jeden Punkt $\vec{x^*}$ im Bild die Position des projizierten Punktes $\vec{x}$ im Fahrzeugkoordinatensystem durch die Gleichungen 3.26 und 3.27 bestimmen.

Das Ausrichten der Betrachtungsbereiche zur Verfolgung der Fahrspurmarkierungen(vgl. Abbildung 3.7) basiert auf den im Fahrzeugkoordinatensystem ermittelten Fahrspurinformationen. Deshalb wird die beschriebene Projektion auch in die andere Richtung, vom Fahrzeugkoordinatensystem in das Bildkoordinatensystem, durchgeführt. Gemäß der bisherigen Notation müssen dazu die Bildpunkte $\vec{x^*}$ und die Fahrzeugkoordinatenpunkte $\vec{x}$ in den Gleichungen 3.26 und 3.27 miteinander vertauscht werden. Die nachfolgend beschriebe Kalibrierung ist für jede Projektionsrichtung einmalig durchzuführen.

3.3.2 Kalibrierung der Transformationsparameter

Die Elemente der Transformationsmatrix **B** (vgl. Gleichung (3.25)) sind zunächst unbekannt und werden durch eine einmalige Kalibrierung berechnet. Die Benutzung der kalibrierten Transformationsmatrix verlangt deshalb ein identisches Ebenenverhältnis zwischen Bild- und Fahrzeugkoordinatensystem wie zum Zeitpunkt der Kalibrierung. Durch Kippen oder Neigen des Fahrzeugs verändert sich das Ebenenverhältnis mit der Folge einer fehlerhaften projektiven Transformation. In Abschnitt 5.2.3 werden ausgewählte Fehlerszenarien untersucht.

Die Kalibrieranordung besteht aus einer Anzahl an Punkten im Fahrzeugkoordinatensytem, in dessen Ursprung das Fahrzeug positioniert wird (vgl. Anhang B). Die genaue Position der Punkte im Fahrzeugkoordinatensystem ist bekannt. Im linsenverzeichnungsfreien Kamerabild werden die korrespondierenden Punkte ausgewählt und ihre Bildkoordinaten notiert. Mit Hilfe der korrespondierenden Punkte wird ein lineares Gleichungssystem aufgestellt. Zusätzlich werden Testpunkte im Fahrzeugkoordinatensystem ausgewählt und deren Korrespondenz im Bild ermittelt. Dadurch lässt sich die korrekte Funktion der projektiven Transformation verifizieren. Die Testpunkte werden nicht in dem Gleichungssystem verwendet. Anhang B skizziert das Vorgehen für die Testumgebung des Prototyps.

Die Gleichungen 3.26 und 3.27 stellen den Ausgangspunkt für die Erstellung des Gleichungssystems dar. Für die Kalibrierung werden die Punkte in kartesischen Koordinaten in die Gleichungen eingesetzt. Durch schrittweise Umstellung lassen sich die unbekannten b_{ij} auf eine Seite bringen:

$$\begin{aligned} & x_k = \frac{x_h}{1} = \frac{b_{11}x_k^* + b_{12}y_k^* + b_{13}}{b_{31}x_k^* + b_{32}y_k^* + 1} \\ \Leftrightarrow \quad & x_k(b_{31}x_k^* + b_{32}y_k^* + 1) = b_{11}x_k^* + b_{12}y_k^* + b_{13} \\ \Leftrightarrow \quad & x_k = b_{11}x_k^* + b_{12}y_k^* + b_{13} - b_{31}x_k^*x_k - b_{32}y_k^*x_k \end{aligned} \tag{3.28}$$

Analog gilt für y_k:

$$\begin{aligned} & y_k = \frac{y_h}{1} = \frac{b_{21}x_k^* + b_{22}y_k^* + b_{23}}{b_{31}x_k^* + b_{32}y_k^* + 1} \\ \Leftrightarrow \quad & y_k(b_{31}x_k^* + b_{32}y_k^* + 1) = b_{21}x_k^* + b_{22}y_k^* + b_{23} \\ \Leftrightarrow \quad & y_k = b_{21}x_k^* + b_{22}y_k^* + b_{23} - b_{31}x_k^*x_k - b_{32}y_k^*x_k \end{aligned} \tag{3.29}$$

Aus den Gleichungen 3.28 und 3.29 lässt sich ein lineares Gleichungssystem in Matrixform erstellen (vgl. Gleichung 3.30). Jedes korrespondierende Punktepaar ergibt zwei Gleichungen, eine für die x-Koordinate und eine für die y-Koordinate. Zur Berechnung der acht unbekannten

b_{ij} werden mindestens acht Gleichungen benötigt. Es werden demnach $n \geq 4$ (mit $n \in \mathbb{N}$) korrespondierende Punkte verwendet.

$$\begin{pmatrix} x^*_{k_1} & y^*_{k_1} & 1 & 0 & 0 & 0 & -x^*_{k_1}x_{k_1} & -y^*_{k_1}x_{k_1} \\ 0 & 0 & 0 & x^*_{k_1} & y^*_{k_1} & 1 & -x^*_{k_1}y_{k_1} & -y^*_{k_1}y_{k_1} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x^*_{k_n} & y^*_{k_n} & 1 & 0 & 0 & 0 & -x^*_{k_n}x_{k_n} & -y^*_{k_n}x_{k_n} \\ 0 & 0 & 0 & x^*_{k_n} & y^*_{k_n} & 1 & -x^*_{k_n}y_{k_n} & -y^*_{k_n}y_{k_n} \end{pmatrix} \cdot \begin{pmatrix} b_{11} \\ b_{12} \\ b_{13} \\ b_{21} \\ b_{22} \\ b_{23} \\ b_{31} \\ b_{32} \end{pmatrix} = \begin{pmatrix} x_{k_1} \\ y_{k_1} \\ \vdots \\ x_{k_n} \\ y_{k_n} \end{pmatrix} \tag{3.30}$$

Bei der Auswahl von vier Punkten $\vec{x^*_i} = (x^*_{k_i} \quad y^*_{k_i})^T$ mit $i \in 0,...,n$ dürfen zur korrekten Berechnung der Projektionsmatrix keine drei $\vec{x^*_i}$ kollinear liegen [Richard Hartley und Andrew Zisserman (2003)]. Für die korrespondierenden Punkte $\vec{x}_i = (x_{k_i} \quad y_{k_i})^T$ muss dieses Kriterium nicht zutreffen.

Um den Fehler durch Kollinearität und Ungenauigkeit bei der Bestimmung der korrespondierenden Koordinaten möglichst gering zu halten, werden mehr als vier Korrespondenzen verwendet. Das entstehende, überbestimmte Gleichungssystem wird zum Erzielen einer hohen Genauigkeit mit der Singulärwertzerlegung gelöst (vgl. Abschnitt 3.4.2). Die Lösung ist die Transformationsmatrix **B** (vgl. Gleichung (3.25)) mit minimalem quadratischen Fehler bezüglich aller ausgewählten Korrespondenzen.

3.4 Identifikation der Polynomparameter durch Ausgleichsrechnung

Der letzte Verfahrensschritt zur Approximation der Fahrspur ist die Schätzung der Polynomparameter des mathematischen Fahrspurmodells (vgl. Abschnitt 2.3) mit Hilfe der bearbeiteten Fahrspurmarkierungspunkte. In der Literatur über numerische Mathematik wird die Parameterschätzung auch als Ausgleichsrechnung bezeichnet [Thomas Huckle und Stefan Schneider (2002), Kapitel 10].

Nachfolgend führt Abschnitt 3.4.1 die numerischen Grundlagen der Ausgleichsrechnung ein. Anschließend wird in Abschnitt 3.4.2 das verwendete Verfahren zur Parameteridentifikation beschrieben.

3.4.1 Numerische Grundlagen der Ausgleichsrechnung

Ausgangspunkt der Ausgleichsrechnung ist eine Messreihe mit m Messpunkten $M_i = (x_i, y_i)$, $i = 1, \cdots, m$, wobei $x_i \neq x_k$ für $i \neq k$ gilt. Die Messpunkte sollen durch eine stetige Funktion f approximiert werden, also $f(x_i) \approx y_i$. In diesem Fall wird die Messreihe durch ein kubisches Polynom mit $n = 4$ Parametern approximiert

$$p(x_i) \approx ax_i^3 + bx_i^2 + cx_i + d, \tag{3.31}$$

wobei $n < m$ gelten soll. Unter dieser Bedingung führt die Approximation der Funktion $p(x_i)$ zu einer Abweichung (vgl. Abbildung 3.10)

$$\begin{aligned} r_i &:= y_i - (ax_i^3 + bx_i^2 + cx_i + d) \\ &:= y_i - p(x_i), \end{aligned} \tag{3.32}$$

die nach einem wählbaren Kriterium minimal sein soll. Üblicherweise wird die Länge des Residuenvektors

$$\vec{r} = \begin{pmatrix} r_1 \\ \vdots \\ r_m \end{pmatrix}, \tag{3.33}$$

also dessen 2-Norm

$$\|\vec{r}\|_2 := \sqrt{\sum_{i=1}^{m} r_i^2} \tag{3.34}$$

als Kriterium angegeben und zur einfacheren Handhabung dessen Quadrat

$$\begin{aligned} \|\vec{r}\|_2^2 &:= \sum_{i=1}^{m} r_i^2 \\ &:= \sum_{i=1}^{m} (y_i - (ax_i^3 + bx_i^2 + cx_i + d))^2 \\ &:= p(x) \end{aligned} \tag{3.35}$$

betrachtet [Norbert Herrmann (2004), Seite 56ff]. Die Minimierung von Gleichung (3.35) wird als *Methode der kleinsten Quadrate* bezeichnet [Martin Hermann (2006), Seite 353ff]. Sie führt zu einer optimalen Funktion p(x) bezüglich der quadratischen Länge des Residuenvektors $\vec{r}$.

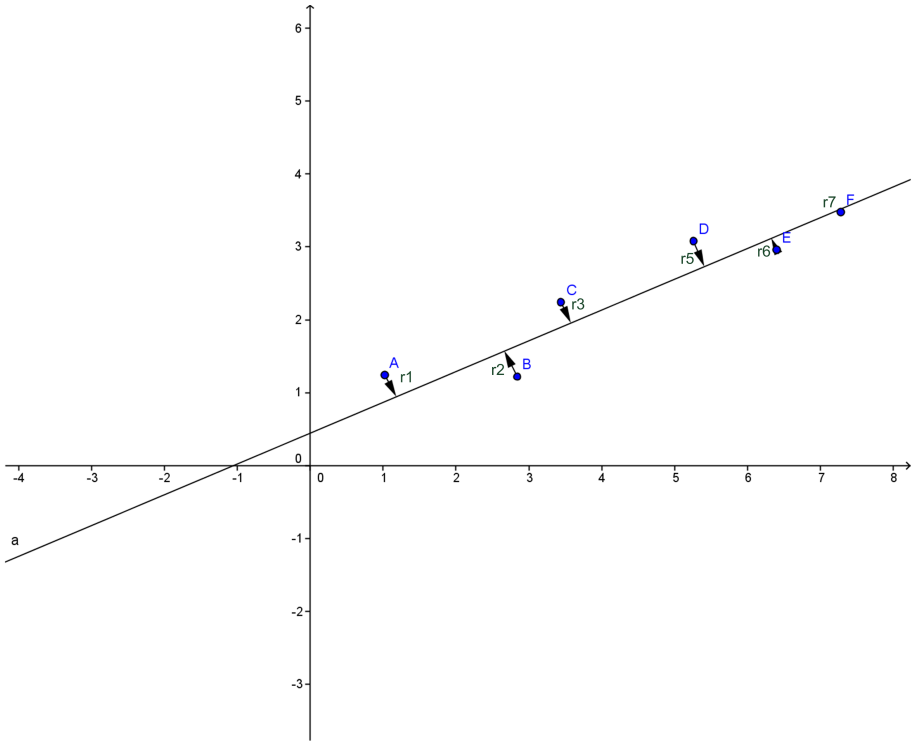


Abb. 3.10: Grafische Veranschaulichung des Abstands r_i einer approximierten Funktion zum Messpunkt (x_i, y_i) am Beispiel einer Geraden

Eine Variante der *Methode der kleinsten Quadrate* ist die Gewichtung der Summanden:

$$p(x) := \sum_{i=1}^{m} g_i \cdot (y_i - (ax_i^3 + bx_i^2 + cx_i + d))^2 \tag{3.36}$$

Diese Variante kann beispielsweise eingesetzt werden, um das kubische Polynom stärker an den nahe dem Fahrzeug liegenden Punkten auszurichten. Das Gewicht g_i ist dann für die nahen Punkte gering (z.B. 0.1) und nimmt proportional zur Entfernung des Punktes zu. Dadurch wird die Summe besonders gering, wenn die nahe liegenden Punkte einen geringen quadratischen Abstand. In dieser Arbeit sind die Gewichte g_i auf 1 festgelegt.

Die Lösung der *Methode der kleinsten Quadrate* (vgl. Gleichung 3.35) wird nachfolgend in Vektorschreibweise angegeben. Aus dem linearen Gleichungssystem mit den unbekannten Parametern a, b, c, d und m Messpunkten (x_i, y_i)

$$\begin{aligned} ax_1^3 + bx_1^2 + cx_1 + d &= y_1 \\ ax_2^3 + bx_2^2 + cx_2 + d &= y_2 \\ &\vdots \\ ax_m^3 + bx_m^2 + cx_m + d &= y_m \end{aligned} \tag{3.37}$$

wird durch Zusammenfassen der bekannten x_i in einer Matrix und der unbekannten a, b, c, d, sowie der bekannten y_i in Spaltenvektoren ein lineares Gleichungssystem in Vektorschreibweise:

$$\mathbf{X} = \begin{pmatrix} x_1^3 & x_1^2 & x_1 & 1 \\ x_2^3 & x_2^2 & x_2 & 1 \\ \vdots & \vdots & \ddots & \vdots \\ x_m^3 & x_m^2 & x_m & 1 \end{pmatrix} \qquad \vec{a} = \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} \qquad \vec{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix} \tag{3.38}$$

Vereinfachend lässt sich dieses System schreiben als:

$$\mathbf{X} \cdot \vec{a} = \vec{y} \tag{3.39}$$

Die Lösung des Systems aus Gleichung (3.39) nach der *Methode der kleinsten Quadrate* ist durch das System der Gaußschen Normalgleichungen

$$\mathbf{X}^T \cdot \mathbf{X}\vec{a} = \mathbf{X}^T \vec{y} \tag{3.40}$$

gegeben [Norbert Herrmann (2004), Seite 57]. Die Umstellung nach $\vec{a}$ führt zu [Martin Hermann (2006), Seite 467]:

$$\vec{a} = \mathbf{X}^\dagger \vec{y} \quad \text{mit} \quad \mathbf{X}^\dagger = \left(\mathbf{X}^T\mathbf{X}\right)^{-1}\mathbf{X}^T \tag{3.41}$$

$\mathbf{X}^\dagger$ wird als Pseudo-Inverse bezeichnet.

Die numerische Berechnung der Pseudoinversen in Gleichung (3.41) führt aufgrund der Berechnung $(\mathbf{X}^T \cdot \mathbf{X})$ zu Fehlern in der Lösung $\vec{a}$ [William H. Press u. a. (2007), Seite 791]. Insbesondere bei größeren Gleichungssystemen, wie der Kalibrierung der projektiven Transformation (vgl. Abschnitt 3.3) wirkt sich dieser Fehler aus. Eine alternative Berechnungsmethode der Pseudo-Inversen ist durch die nachfolgend beschriebene Singulärwertzerlegung gegeben. Aus Gründen der Einfachheit wird sie ebenfalls zum Lösen der Polynomapproximation verwendet.

3.4.2 Anwendung der Singulärwertzerlegung

Die Singulärwertzerlegung teilt die *m x n* Matrix **X** in drei Matrizen **U**, **W** und **V** auf:

$$\mathbf{X} = \mathbf{U} \cdot \mathbf{W} \cdot \mathbf{V}^T \tag{3.42}$$

U ist eine orthogonale *m x n* Matrix, **V** ist eine orthogonale *n x n* Matrix und W ist eine *n x n* Diagonalmatrix [William H. Press u. a. (2007), Abschnitt 2.6]. Für eine orthogonale Matrix **O** gilt:

$$\mathbf{O}^T\mathbf{O} = \mathbf{I} \quad \text{und} \quad \mathbf{O}^T = \mathbf{O}^{-1} \tag{3.43}$$

Statt der numerisch ungünstigen Inversion von **U** und **V** kann nach Gleichung (3.43) jeweils die transponierte Matrix gebildet werden. Dieser Sachverhalt findet in Gleichung (3.44) Anwendung. Die Diagonalmatrix **W** enthält die positiven Singulärwerte w_i mit $i = 1, \cdots, n$ von **X**.

Für den Fall $m > n$ gilt für die Singulärwertzerlegung [William H. Press u. a. (2007), Abschnitte 2.6.2 und 2.6.4]

$$\mathbf{X}^{-1} = \mathbf{V} \cdot \mathbf{W}^{-1} \cdot \mathbf{U}^T, \tag{3.44}$$

wobei für die Elemente der inversen Diagonalmatrix **W**

$$w_i = \frac{1}{w_i} \tag{3.45}$$

gilt. Wird der Kehrwert $\frac{1}{w_i}$ von einem w_i nahe 0 ebenfalls auf 0, statt auf ∞ gesetzt

$$w_i = \begin{cases} \frac{1}{w_i}, & \text{falls } w_i \neq 0, \\ 0, & \text{falls } w_i \approx 0. \end{cases} \tag{3.46}$$

so entspricht Gleichung (3.44) der Pseudo-Inversen [Martin Hermann (2006), Seite 483ff]:

$$\mathbf{X}^\dagger = \mathbf{V} \cdot \mathbf{W}^\dagger \cdot \mathbf{U}^T \tag{3.47}$$

Durch einsetzten von Gleichung (3.47) in Gleichung (3.41) ergibt sich die Lösung nach der *Methode der kleinsten Quadrate*:

$$\vec{a} = \mathbf{V} \cdot \mathbf{W}^\dagger \cdot \mathbf{U}^T \cdot \vec{y} \tag{3.48}$$

In der softwaretechnischen Umsetzung der Identifikation der Polynomparameter mit der Singulärwertzerlegung wird auf die Funktion *cvSolve(...)* aus der OpenCV Grafikbibliothek [opencv (2006)] zurück gegriffen. Es reicht dazu aus, die Matrix **X**, sowie den Vektor $\vec{y}$ anzugeben und die Lösung $\vec{a}$ durch *cvSolve(...)* berechnen zu lassen.

4 Identifikation von Fahrzeuglage, Fahrzeugsteuerkurs und Fahrspurradius

Anhand der kubischen Ausgleichspolynome (vgl. Abschnitt 2.3) für die linke und rechte Fahrspurmarkierung lassen sich Berechnungen zur Ermittlung der Fahrzeuglage, des Fahrzeugkurses und des Fahrspurverlaufs durchführen. Die Auswertung der Ausgleichpolynome erfolgt im Fahrzeugkoordinatensystem (vgl. Abschnitt 2.2). In den nachfolgenden Abschnitten dieses Kapitels wird die Berechnung folgender Messgrößen beschrieben:

- der laterale Abstand A_h zwischen dem Ursprung des Fahrzeugkoordinatensystems und der Fahrspurmitte,
- der Steuerkurswinkel ψ des Fahrzeugs bezogen auf die Fahrspur,
- sowie der Kurvenradius R_k und die Kurvenposition P_k, falls die Fahrspur im Sichtbereich eine Kurve enthält.

Die nachfolgenden Abbildungen (vgl. 4.1, 4.2 und 4.3) veranschaulichen die Berechnungen am Beispiel einer Kurve.

4.1 Fahrzeuglage als lateraler Abstand zur Fahrspurmitte

Der laterale Abstand A_h zur Fahrspurmitte kann von der Spurführungsregelung des Fahrzeugs verwendet werden, um die maximale Entfernung des Fahrzeugs zu beiden Fahrpurmarkierungen einzustellen. Der laterale Abstand der Fahrspurmitte bezieht sich auf den Ursprung des Fahrzeugkoordinatensystems.

Zur Berechnung der Fahrspurbreite b werden die Schnittpunkte S_l und S_r der Approximationspolynome $p_l(x)$ und $p_r(x)$ der linken und rechten Fahrspurmarkierungen mit der y-Achse gebildet (vgl. Abbildung 4.1). Die Breite ergibt sich aus der Differenz der y-Koordinaten der Schnittpunkte:

$$b = \Delta y = y_{S_r} - y_{S_l} \quad \text{mit} \quad y_{S_l} = p_l(0) \quad \text{und} \quad y_{S_r} = p_r(0) \tag{4.1}$$

Die y-Koordinate der Fahrspurmitte m entspricht in diesem Fall dem Abstand zum Ursprung des Fahrzeugkoordinatensystems:

$$A_h = -1 * y_m \quad \text{mit} \quad y_m = \frac{b}{2} \tag{4.2}$$

Das Vorzeichen wird umgedreht, damit der laterale Abstand nach links das negative Vorzeichen erhält (vgl. Abbildung 4.1).

Bei einem hohen Steuerkurswinkel (vgl. Abschnitt 4.2) oder einer zu großen Störung der Bilddaten (vgl. Abschnitt 3.1.2, Gleichung 3.3) liegen nicht genug Informationen zur Approximation mit Ausgleichspolynomen vor. Wenn $p_l(x)$ oder $p_r(x)$ nicht mehr gebildet werden können, legt das Verfahren die größtmögliche, ganzzahlige Maschinenzahl (*INT_MAX*) als horizontalen Abstand fest, um den Fehlerfall zu signalisieren.

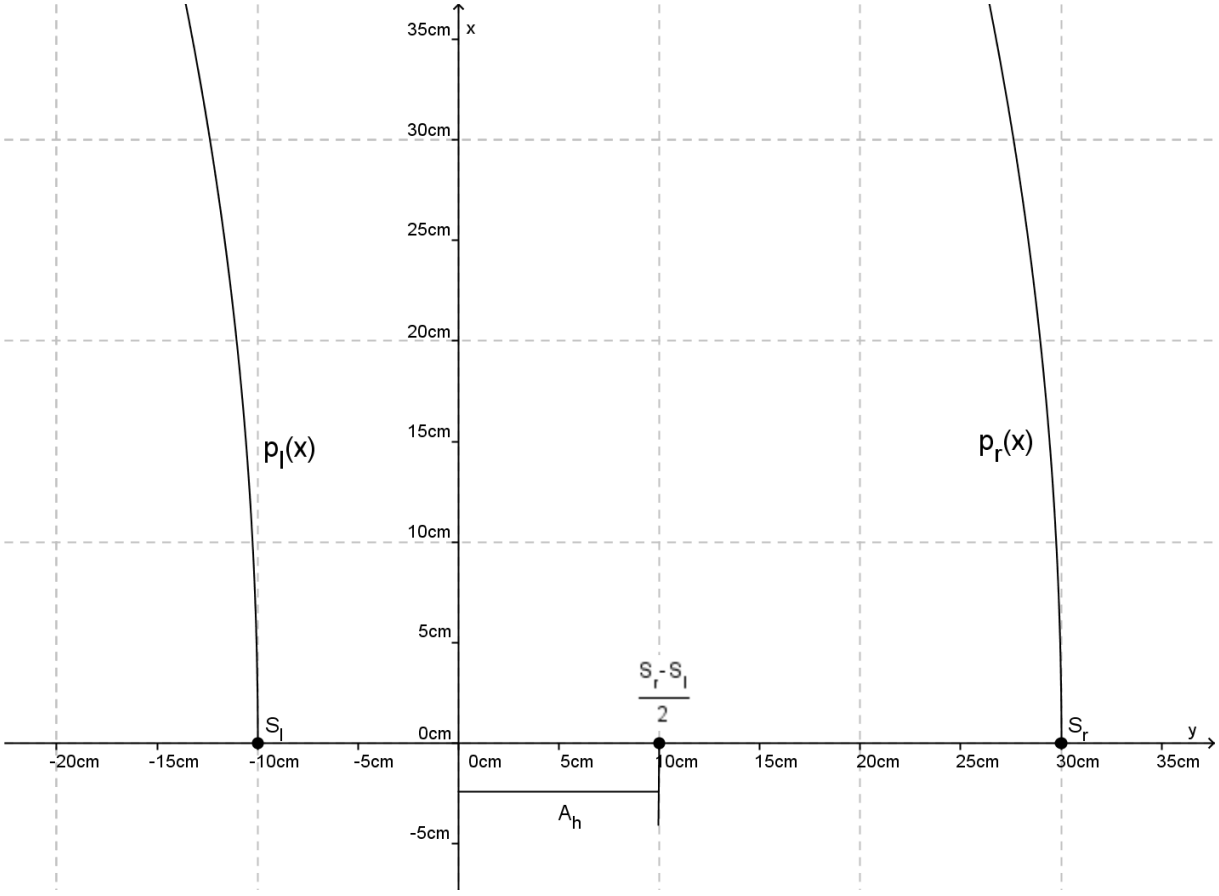


Abb. 4.1: Berechnung des lateralen Abstands A_h zum Ursprung des Fahrzeugkoordinatensystems

4.2 Steuerkurswinkel des Fahrzeugs zur Fahrspur

Der Steuerkurswinkel ψ gibt die Abweichung des Fahrzeugsteuerkurses von der Richtung der Fahrspur in Grad an. Gemessen wird der Steuerkurswinkel an der Stelle $x = 0$ im Fahrzeugkoordinatensystem (vgl. Abbildung 4.2). Ein negativer Winkel soll einer Abweichung nach links, ein positiver Winkel einer Abweichung nach rechts entsprechen. Die Berechnung wird an einem

ausgewählten Ausgleichspolynom durchgeführt und muss gegebenfalls für weitere Polynome wiederholt werden.

Die erste Ableitung gibt die Richtung des kubischen Polynoms $p(x) = ax^3 + bx^2 + cx + d$ (vgl. Abschnitt 2.3) an:

$$p'(x) = 3ax^2 + 2bx + c \tag{4.3}$$

Durch die Messung an der Stelle $x = 0$ vereinfacht sich Gleichung (4.3) zu

$$p'(0) = c \tag{4.4}$$

Der Steuerkurswinkel ψ wird tangential aus der Polynomrichtung gebildet (vgl. Abbildung 4.2):

$$\psi = -1 * \arctan(c) \tag{4.5}$$

Das Ergebnisvorzeichen des Arkustangens muss umgekehrt werden, um die genannten Anforderungen zu erfüllen (vgl. Abschnitt 1.3). Ein positiver Steuerkurswinkel ψ signalisiert somit ein bevorstehendes Verlassen der Fahrspur nach rechts.

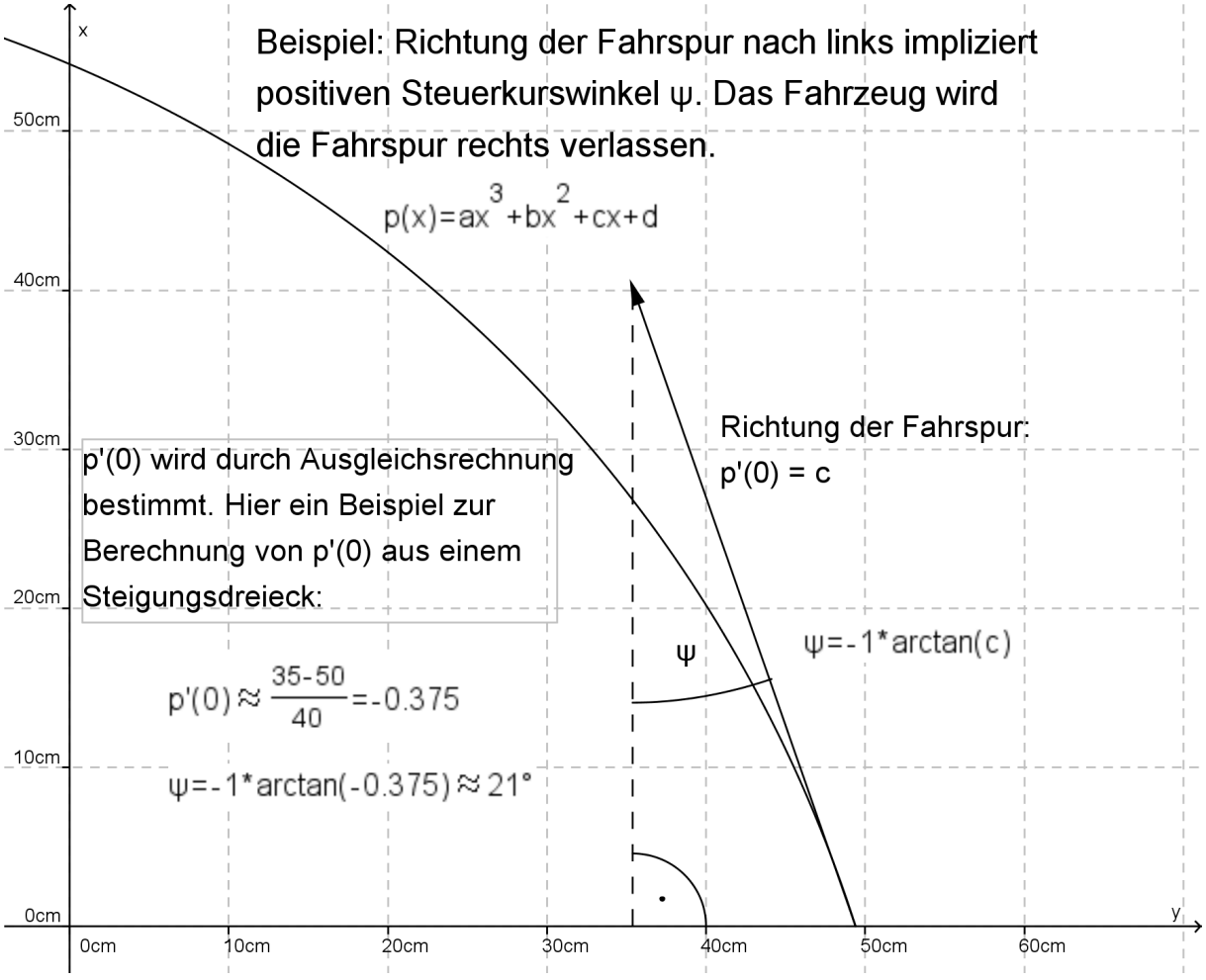


Abb. 4.2: Berechnung des Steuerkurswinkels bezogen auf ein Ausgleichspolynom

Aufgrund der wenigen Bewegungsgrade des kubischen Polynoms (vgl. Abschnitt 5.2.4) ist die Messung des Steuerkurswinkels fehlerbehaftet . Der Einfluss der Polynomkrümmung auf den Steuerkurswinkel wird in Abschnitt 5.3 beschrieben.

4.3 Kurvenradius der Fahrspur

Ein Polynom 3. Grades stellt nur bei geringen Krümmungen der Fahrspur eine gute Annäherung eines Kreises zur Bestimmung des Fahrspurradius dar (vgl. Abschnitt 2.3, Abbildung 2.2).

Unter Berücksichtigung der Eigenschaft, dass ein Polynom an unterschiedlichen Stellen, unterschiedliche Krümmungskreise besitzt (vgl. Abbildung 4.3), lässt sich dennoch eine positionsbezogene Information über den Radius der Fahrspur extrahieren. Die Regelungssoftware entscheidet selbstständig an welcher Stelle der Radius gemessen werden soll.

Die Berechnung der Krümmung C eines Polynoms $p(x)$ an einer Stelle x ist durch

$$C = \frac{p''(x)}{\sqrt{1+(p'(x))^2}^3} \tag{4.6}$$

gegeben [Lothar Papula (2007), Abschnitt 3.3.2]. Für den Radius $r = \frac{1}{C}$ folgt daraus

$$r = \frac{\sqrt{1+(p'(x))^2}^3}{p''(x)} \tag{4.7}$$

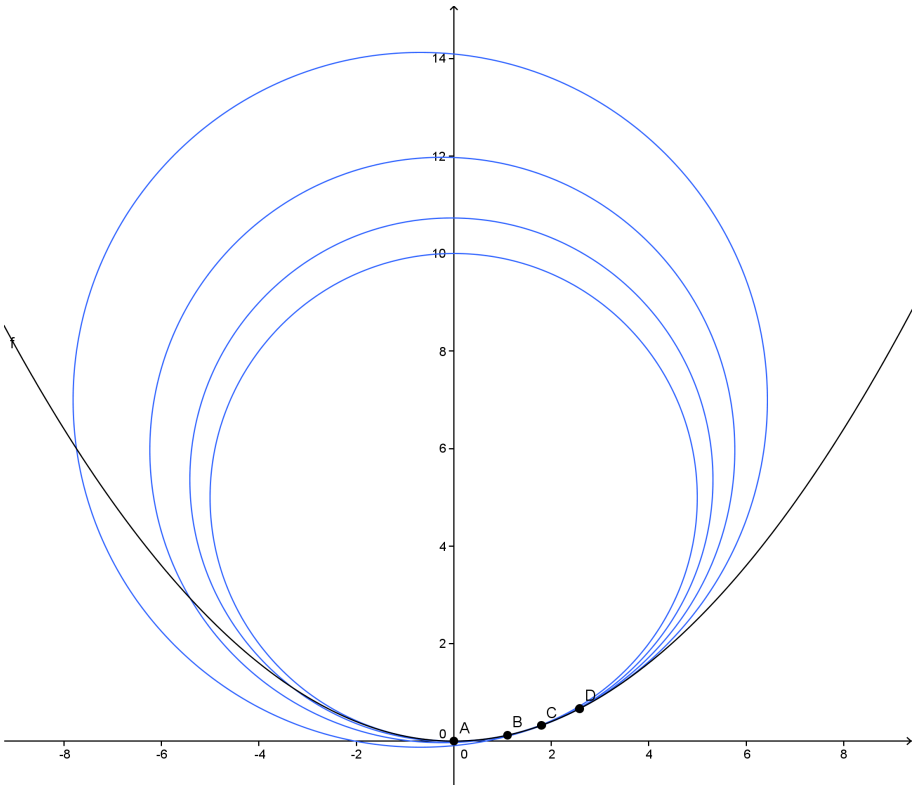


Abb. 4.3: Krümmungskreise eines quadratischen Polynoms zu den Punkten A,B,C und D

5 Messtechnische Auswertung der Approximation und Identifikation

In diesem Kapitel erfolgt die messtechnische Auswertung der Verfahrensschritte zur Approximation des Fahrspurmodells und zur Identifikation der Fahrzeuglage, Fahrzeugorientierung und des Fahrspurradius.

Zunächst wird in Abschnitt 5.1 die eingesetzte Testumgebung spezifiziert. Abschnitt 5.2 beinhaltet die Auswertung der Verfahrensschritte zur Approximation. Der Einfluss von Fehlern durch die Approximation auf die Systemidentifikation wird in Abschnitt 5.3 erläutert. Zuletzt fasst Abschnitt 5.4 die Rechenzeiten der einzelnen Verfahrensschritte in der Testumgebung tabellarisch zusammen.

5.1 Beschreibung der Testumgebung

Vor der Auswertung der einzelnen Verfahrensschritte wird zunächst die eingesetzte Hardware- und Softwareumgebung zur Durchführung der Messungen beschrieben.

Die Hauptrecheneinheit der Testumgebung besteht aus einem Embedded-PC und einer Flash-Festplatte (SSD - Solid State Disk). Die Leistungsdaten des Embedded-PC werden in Tabelle 5.1 dargestellt. Die eingesetzte Kamera zeichnet Bilder in einem breiten Format auf und ist mit einem Weitwinkelobjektiv ausgestattet (vgl. Tabelle 5.2). Das Weitwinkelobjektiv ermöglicht eine Erkennung der Fahrspurmarkierungen in engen Kurven (vgl. [Richard Matthaei (2007)]). Die kalibrierten internen Kameraparameter sind in Anhang A aufgeführt.

Angebracht sind der Embedded-PC und die Kamera auf einem Modellfahrzeug im Maßstab 1:10 (vgl. Abbildung 5.1 und Tabelle 5.3). Das Modellfahrzeug ist neben der Kamera mit Ultraschall- und Infrarotsensoren, sowie Inkrementalgebern zur Strecken- und Geschwindigkeitsmessung ausgestattet (vgl. [Eike Jenning (2008b)], Anhang A.1). Für die Fahrspurerkennung werden der Lenkwinkel und die Geschwindigkeit ausgewertet (vgl. Abschnitt 2.4). Der Lenkwinkel wird in der Testumgebung nicht sensorisch erfasst. Alle Sensorwerte werden dem PC im 50 Millisekundentakt zur Verfügung gestellt.

Der Einsatz des erarbeiteten Verfahrens zur Fahrspurerkennung und Systemidentifikation auf dem Modellfahrzeug erfordert eine Kalibrierung der internen Parameter der eingesetzten Kamera (vgl. Abschnitt 3.2 und Anhang A), sowie eine Kalibrierung der projektiven Transformation (vgl. Abschnitt 3.3 und Anhang B).

Die beschriebene Testumgebung wurde verwendet, um eine wettkampfkonforme (vgl. [Carolo-Cup-Regelwerk (2009)]) Teststrecke aufzuzeichnen (vgl. Abbildung 3.8 für ein Beispiel).

Tabelle 5.1: Leistungsdaten der eingesetzten Hauptrecheneinheit

Via Nano-ITX Embedded-PC	
Komponente	
Prozessor	1.0 GHz Via Luke CoreFusion
Arbeitsspeicher	512 MB
Festplatte	8 GB SSD
Betriebssystem	Debian 4.0 mit 2.6.22.1 Echtzeit-Kernel

Tabelle 5.2: Daten der eingesetzten Kamera

IDS uEye1226LE-M	
Merkmale	
Bildauflösung	752 x 480 Pixel
Maximale Bildrate	87 fps bei Vollauflösung
Belichtungszeit	80 μs - 5,5 s
Bildsensor	1/3“ CMOS monochrom
Datenschnittstelle	USB 2.0
Abmessungen	B: 36 mm, H: 36 mm, T: 20 mm
Gewicht	12 g
Objektiv	Weitwinkelobjektiv: f = 2,27 mm, 1/3“

Die Auswertung der Verfahrensschritte wird nachfolgend mit der aufgenommenen Teststrecke durchgeführt.

Für die Testsoftware werden die einzelnen Verfahrenschritte in C++ Klassen gekapselt und durch eine Klasse zur Darstellung des Fahrzeugkoordinatensystems ergänzt (vgl. Anhang D). Der modulare Aufbau ermöglicht die einfache Durchführung unterschiedlicher Testszenarien. In den nachfolgenden Abschnitten werden jeweils die zur Auswertung benutzten Klassen kurz erwähnt und ihre Verwendung aufgezeigt.

Die bereits vorhandene C++ Systemsoftware *FAUSTcore* stellt abstrakte Basiskomponenten zur Implementierung von Treibern und Tasks zur Verfügung (vgl. Anhang C). Des Weiteren bietet der *FAUSTcore* die bequeme Steuerung der Testsoftware durch Parametereinstellungen über eine Web-Oberfläche. Die Testsoftware wird durch die von Task abgeleitete Klasse LaneDetection (vgl. Anhang D) in den *FAUSTcore* integriert und auf dem Embedded-PC ausgeführt.

Tabelle 5.3: Daten des eingesetzten Modellfahrzeugs

1:10 Modellfahrzeug Tamiya „VW Race Touareg“	
Merkmale	
Geschwindigkeit	maximal ca. 10 m/s (geschätzt)
Lenkwinkel	maximal ca. 22°
Radstand	27 cm
Abmessungen	L: 45 cm, B: 19 cm

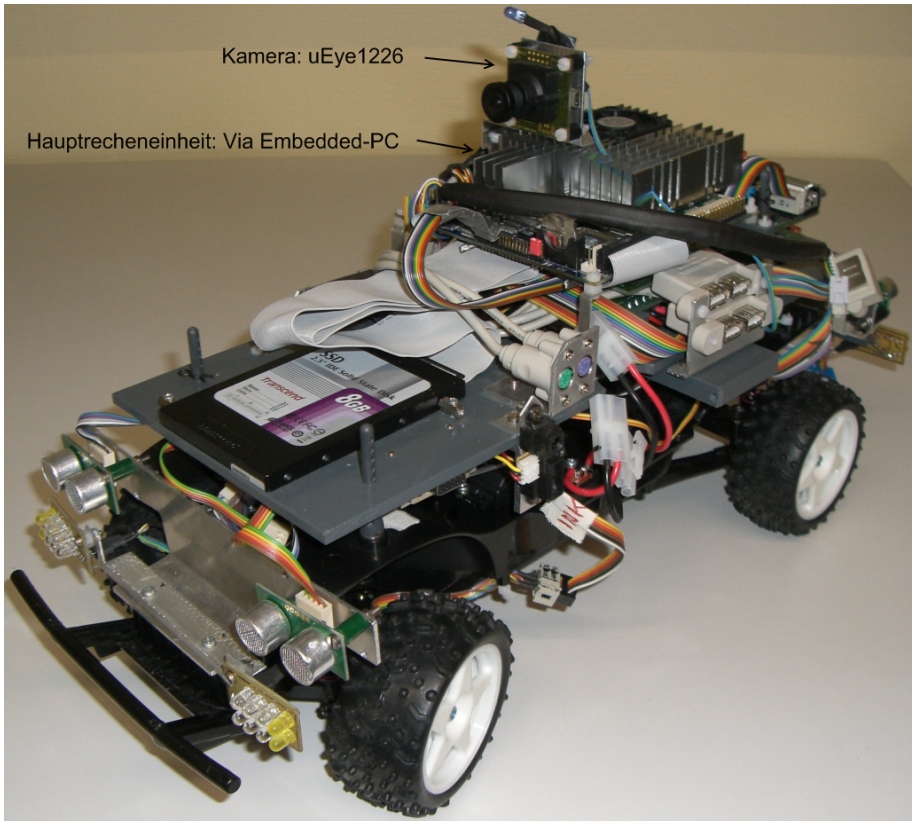


Abb. 5.1: Testumgebung bestehend aus einem Modellfahrzeug mit Kamera und Embedded-PC

5.2 Auswertung der Verfahrensschritte der Approximation

In diesem Abschnitt werden die Verfahrensschritte zur Approximation des Fahrspurmodells ausgewertet. Dazu werden die Genauigkeiten der Linsenverzeichnungskorrektur, der projektiven Transformation und der Ausgleichsrechnung untersucht.

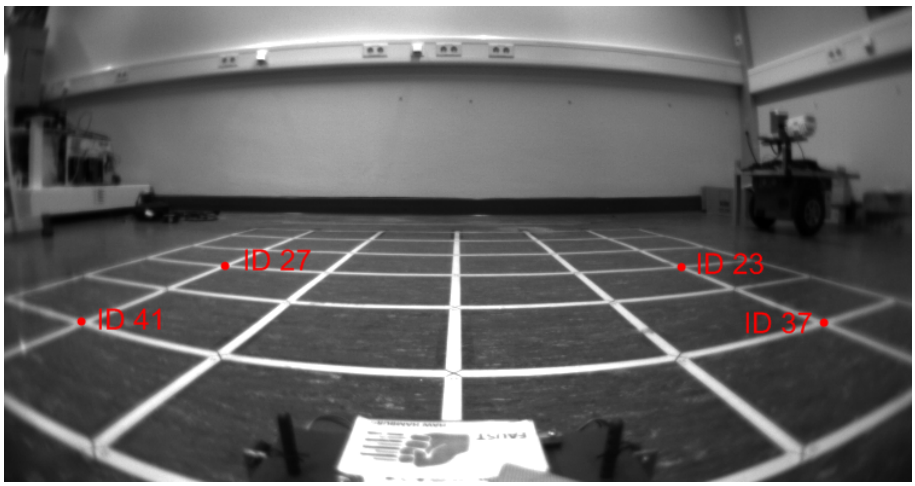
5.2.1 Genauigkeit der Linsenverzeichnungskorrektur

In diesem Abschnitt wird die Genauigkeit der Linsenverzeichnungskorrektur untersucht. Die Auswertung erfolgt anhand selektierter Testpunkte von einer Aufnahme des Kalibriermusters der projektiven Transformation (vgl. Abbildung 5.2 und Anhang B).

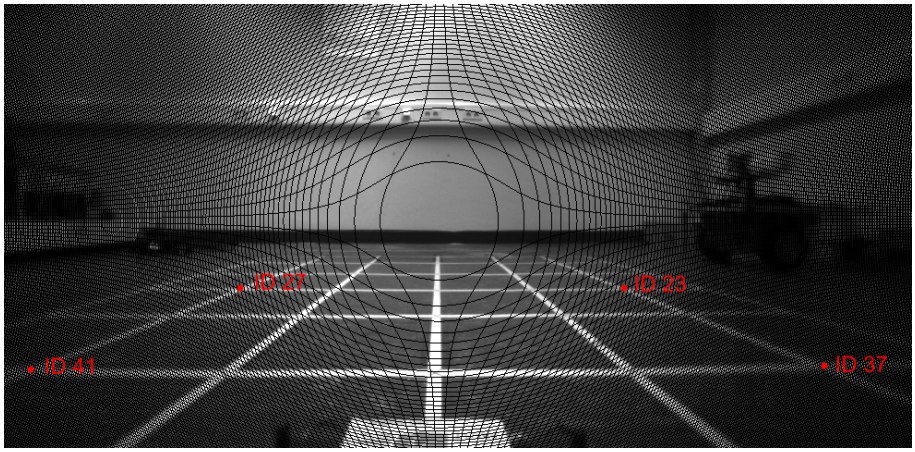
Eine Testsoftware führt die Linsenverzeichnungskorrektur für das gesamte Ausgangsbild (vgl. 5.2a) durch und speichert das Ergebnisbild anschließend ab (vgl. Listing D.1). Die verwendeten Klassen *CameraImage*, *CalibratedUeye1226* und *LensDistortionCorrection* sind in Anhang D beschrieben.

In dem verzeichenten und dem korrespondierenden linsenverzeichnungsfreien Bild werden zur Auswertung Kreuzungspunkte des Testmusters selektiert (vgl. Abbildung 5.2) und dessen Pixelpositionen betrachtet. Die ID-Nummern der Punkte beziehen sich auf Abbildung B.3.

Das Ergebnis der Messung für das verzeichnete Bild vom Testmuster wird in Tabelle 5.4 dargestellt. Da jeweils die linken und rechten selektierten Messpunkte die gleiche Höhe (y-



(a) Verzeichnetes Bild des Testmusters mit den markierten Testpunkten



(b) Linsenverzeichnungsfreies Bild des Testmuster mit den markierten Testpunkten

Abb. 5.2: Vergleich zwischen verzeichnetem und verzeichnungsfreiem Bild des Testmusters

Koordinate) im Bild aufweisen, wird dieses Verhältnis auch für das verzeichnungsfreie Bild erwartet.

Tabelle 5.4: Pixelkoordinaten der selektierten Messpunkte im verzeichneten Bild

Pixelkoordinaten	
ID-Nummer	[x,y]
23	[560,300]
27	[182,300]
37	[679,347]
41	[64,347]

Im verzeichnungsfreien Bild liegen die Messpunkte mit den ID-Nummern 37 und 41 nicht mehr auf gleicher Höhe (vgl. Tabelle 5.5). Die Differenz von 5 Pixeln ist auf die Linsenverzeichnungskorrektur zurückzuführen. Beide Messpunkte liegen in einem stark verzeichneten Bereich des Bildes (vgl. Abbildung 5.2a), sodass sich die Verzeichnungskorrektur dort stark auswirkt. Die Differenz entspricht ungefähr einer Abweichung von 1-2 Zentimetern auf dem Kalibriermuster.

Tabelle 5.5: Pixelkoordinaten der selektierten Messpunkte im verzeichnungsfreien Bild

	Pixelkoordinaten
ID-Nummer	[x,y]
23	[672,307]
27	[256,307]
37	[888,391]
41	[28,396]

Die Ursache für die Ungenauigkeit liegt in der Kalibrierung der internen Kameraparameter begründet. Die Aufnahmen des Photomodeler-Kalibriermusters (vgl. Anhang A) sollten nach Möglichkeit alle Ecken des Kamerabildes einmal abdecken, da dort die Linsenverzeichnung am stärksten ist. Andernfalls werden die Verzeichnungsparameter nicht korrekt bestimmt. Dies sollte bei der Kalibrierung der internen Kameraparameter vor dem Wettbewerb berücksichtigt werden. Für diese Arbeit wird der Fehler aufgrund der geringen Abweichungen toleriert.

5.2.2 Genauigkeit der projektiven Transformation

Zur Kalibrierung der projektiven Transformation zwischen Bild- und Fahrzeugkoordinatensystem wird das Fahrzeug vor dem vermessenen Kalibriermuster aufgestellt (vgl. Anhang B) und ein linsenverzeichnungsfreies Bild aufgenommen. Die Korrespondenzen zwischen Bildpunkten und Kalibriermusterpunkten sind in Abbildung B.3 hinterlegt. Zur Bestimmung der Projektionsmatrix werden alle Korrespondenzen in ein überbestimmtes Gleichungssystem eingesetzt (vgl. Abschnitt 3.3) und daraus die Lösung durch die *Methode der kleinsten Quadrate* berechnet (vgl. Abschnitt 3.4). Die Korrespondenzen (vgl. Abbildung B.2) werden zur softwaretechnischen Umsetzung der Kalibrierung in der Klasse *CaroloCarProjectiveCalibration* manuell eingetragen. Die Kalibrierung der zur Projektion verwendeten Klasse *ProjectiveTransformation* wird in Listing D.2 gezeigt.

Die Güte der Kalibrierung wird mit acht ausgewählten Korrespondenzen geprüft (vgl. Abbildung 5.3). Die Bildpunkte der Korrespondenzen werden zum Test projektiv transformiert und anschließend die Abweichung gegenüber den Koordinaten des korrespondierenden Kalibriermusterpunkts gemessen (vgl. Listing D.3).

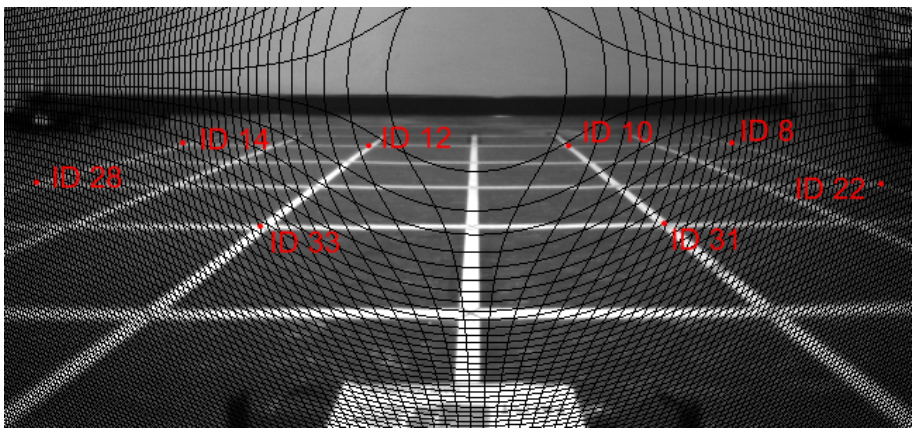


Abb. 5.3: Testpunkte zur Messung der Genauigkeit der projektiven Transformation

Die Ergebnisse der Messung werden in Tabelle 5.6 aufgeführt. Gut zu erkennen ist die Abweichung des Messpunkts mit der ID-Nummer 14 vom Referenzpunkt um etwa 3 Zentimeter. Die anderen Messpunkte sind hingegen bei entsprechender Rundung auf einen Zentimeter genau.

Tabelle 5.6: Abweichungen der Messpunkte zu den Referenzpunkten nach der projektiven Transformation

Abweichungen in cm	
ID-Nummer	[x,y]
8	[-1.13,0.33]
10	[1.37,-0.10]
12	[0.81,-0.69]
14	[-3.10,-2.51]
22	[-0.98,0.07]
28	[-1.47,-0.98]
31	[0.65,-0.04]
33	[0.45,-0.43]

In mehreren Tests mit variierenden Korrespondenzen zur Kalibrierung weist der Messpunkt mit der ID-Nummer 14 stets die größte Abweichung aller Messpunkte auf, selbst wenn er in die Kalibrierung mit einbezogen wird. Die umliegenden Messpunkte mit den ID-Nummern 7, 13 und 21 (vgl. Abbildung B.3) zeigen jeweils Abweichungen von etwa 1 Zentimeter (vgl. Tabelle 5.7).

Tabelle 5.7: Vergleich der Abweichungen der Messpunkte um ID-Nummer 14

Abweichungen in cm	
ID-Nummer	[x,y]
6	[1.39,-1.0]
7	[0.85,-1.31]
13	[0.48,-1.33]
14	[-3.10,-2.51]
20	[-1.01,-0.72]
21	[-1.35,-0.3]

Eine Erklärung für die Ungenauigkeit dieses Punktes ist noch nicht gefunden und muss nachhaltig untersucht werden, bevor das Verfahren Produktstatus erlangt.

5.2.3 Einfluss der Fahrzeugneigung auf die projektive Transformation

Eine Neigung des Modellfahrzeugs während der Fahrt führt zu Veränderungen in dem Verhältnis zwischen der Bildebene und der Weltebene (vgl. Abbildung 3.9). Durch die Kalibrierung der projektiven Transformation ist das Verhältnis der Ebenen zum Kalibrierzeitpunkt fest in der Projektionsmatrix (vgl. Abschnitt 3.3) eingespeichert. Aus diesem Grund führt die Fahrzeugneigung zu einem Fehler in der projektiven Transformation. Nachfolgend wird beispielhaft eine Fahrzeugneigung nach vorne betrachtet.

Für den Test wird das Modellfahrzeug wie zuvor an dem Kalibriermuster aufgestellt (vgl. Abschnitt 5.2.2) und maximal so nach vorne geneigt, dass alle Räder noch kontakt zum Boden haben. Das resultierende Kamerabild wird aufgezeichnet und abgespeichert. Nach der Korrektur der Linsenverzeichnung (vgl. Listing D.1 und Abbildung 5.4) erfolgt ein Vergleich mit dem linsenverzeichnungsfreien Referenzbild (vgl. u.a. Abbildung 5.3) ohne Fahrzeugneigung.

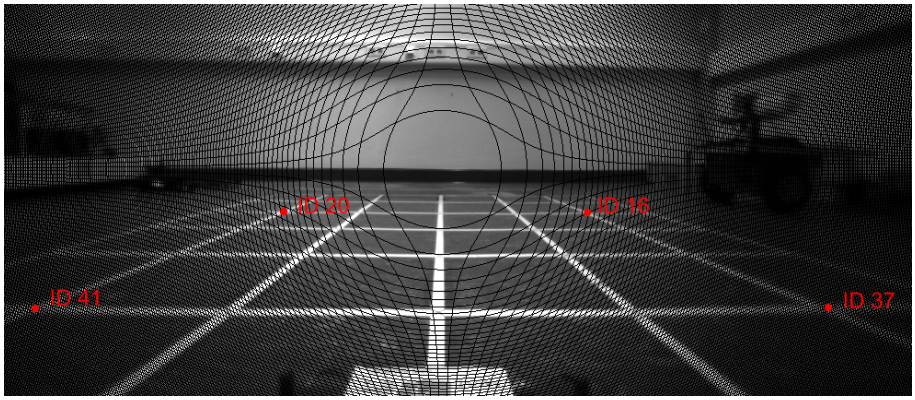


Abb. 5.4: Lage der Messpunkte bei Neigung des Fahrzeugs nach vorne

Durch die Neigung nach vorne wird gegenüber dem Referenzbild eine Lageveränderung der Bildpunkte nach oben erwartet. Für den Vergleich werden einige Messpunkte definiert (vgl. Abbildung 5.4) und deren Veschiebung gemessen (vgl. Tabelle 5.8).

Tabelle 5.8: Einfluss einer Fahrzeugneigung nach vorne auf die Pixelposition der Messpunkte

Lageveränderung in Pixeln	
ID-Nummer	[x,y]
16	[0,-9]
20	[0,-11]
37	[3,-8]
41	[-2,-10]

Da die y-Werte im Bildkoordinatensystem nach unten abgetragen werden, bedeutet das negative Vorzeichen die erwartete Lageveränderung nach oben. Die Messpunkte mit den ID-Nummern 20 und 41 liegen links im Bild und weisen eine geringfügig höhere Verschiebung als die Messpunkte mit den ID-Nummern 16 und 37 auf. Dies ist auf die nicht ganz exakte Neigung des Fahrzeugs nach vorne zurückzuführen. Die im unteren Bildbereich liegenden Messpunkte mit den ID-Nummern 37 und 41 weisen aus diesem Grund zusätzlich eine geringe horizontale Abweichung auf.

Die Verschiebung der Messpunkte um etwa 10 Pixel wirkt sich bei der projektiven Transformation insbesondere in Bildbereichen nahe des Horizonts aus (vgl. Tabelle 5.9). Beim Betrachten der Tabelle ist die geänderte Achsenzuordnung des Kalibriermusters gegenüber dem verzeichnungsfreien Bild zu beachten (vgl. Abbildung B.3).

Die Messpunkte mit den ID-Nummern 16 und 20 liegen um mindestens 20 Zentimeter weiter vom Ursprung des Kalibriermusters entfernt, als deren Referenzmesswerte ohne Fahrzeugneigung. Zudem ist eine seitliche Lageveränderung um jeweils etwa zehn Zentimeter nach außen zu

Tabelle 5.9: Einfluss einer Fahrzeugneigung nach vorne auf die Position der Messpunkte nach der projektiven Transformation

Lageveränderung in Zentimetern	
ID-Nummer	[x,y]
16	[22.13,10.60]
20	[29.04,-14.05]
37	[2.19,1.89]
41	[3.5,-5.32]

beobachten. Die Konsequenz dieser Lageveränderungen ist die Abbildung der parallel liegenden Messpunkte (vgl. Abbildung B.3) in einen nach oben geöffneten Trichter (vgl. Abbildung 5.5).

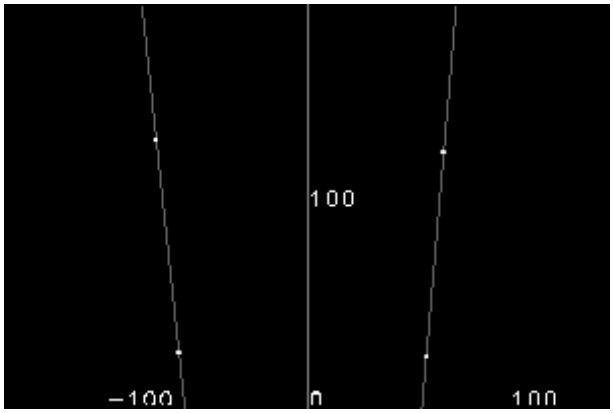
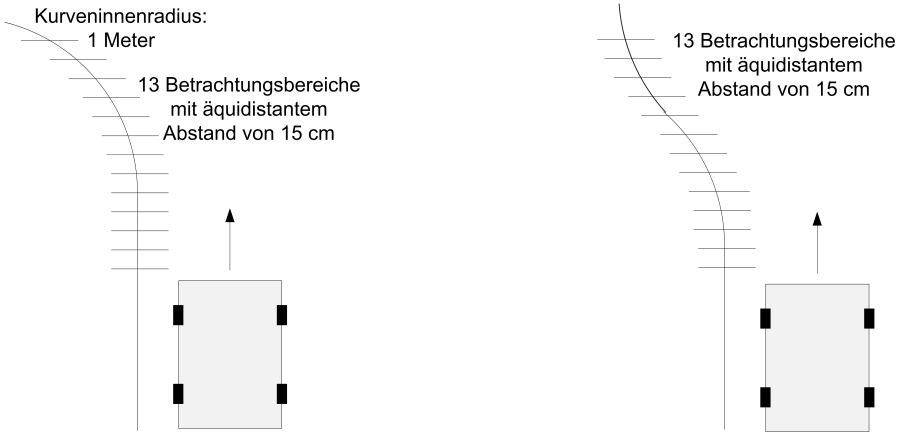


Abb. 5.5: Projektive Transformation von parallelen Geraden zu einem Trichter bei Neigung des Fahrzeugs nach vorne

Der in diesem Abschnitt beschriebene Fehler ist eine Konsequenz aus den Fahrzeugbewegungen beim Beschleunigen, Bremsen und Lenken. Da diese Bewegungen selbst bei einer steifen Federung des Fahrzeugs nicht auszuschließen sind, sollte die resultierende Fahrzeugneigung sensorisch erfasst und in die Projektion zur Laufzeit eingerechnet werden (vgl. Kapitel 7).

5.2.4 Approximationsgenauigkeit eines Polynoms bei ungestörten Messwerten

In diesem Abschnitt wird die Approximationsgenauigkeit von Polynomen 2. und 3. Grades bei ungestörten Messwerten untersucht. Für die Tests wird eine gerade Zufahrt des Fahrzeugs auf eine Kurve mit dem minimalen Innenradius von 1 Meter (vgl. [Carolo-Cup-Regelwerk (2009)]) und die gerade Zufahrt auf eine S-Kurve simuliert (vgl. Abbildung 5.6). Die Simulation erfolgt durch das Einzeichnen der imaginären Schnittpunkte der Fahrspurmarkierung mit den Betrachtungsbereichen (vgl. Abbildung 5.6) in das Fahrzeugkoordinatensystem. Mit jedem Bewegungsschritt bewegt sich das Fahrzeug um 15 Zentimeter auf die Kurve beziehungsweise S-Kurve zu. Der Test wird abgeschlossen, sobald das Fahrzeug direkt am Kurvenbeginn steht.



(a) Testszenario: Zufahrt auf Kurve mit Innenradius von 1 Meter

(b) Testszenario: Zufahrt auf S-Kurve

Abb. 5.6: Testszenarien zur Auswertung der Approximationsgenauigkeit von Polynomen 2. und 3. Grades

Die Software verwendet zur Implementierung der Tests die Klassen *CarCoordinateSystem* und *Polynomial* (vgl. Anhang D). Die Linsenverzeichnungskorrektur und die projektive Transformation werden nicht benötigt, da die Fahrspurmarkierungspunkte direkt in das Fahrzeugkoordinatensystem (vgl. Abschnitt 2.2) eingezeichnet werden. Ein Anwendungsbeispiel der verwendeten Klassen zeigt Listing D.4.

Die Ergebnisbilder der Simulation sind aufgrund des Umfangs in Anhang F hinterlegt. Zur Auswertung der Approximationsgenauigkeit wird jeweils die 2-Norm der Residuenvektoren (vgl. Abschnitt 3.4) der Polynome 2. und 3. Grades verglichen.

Gerade Zufahrt auf eine Kurve mit minimalem Radius von 1 Meter

Die gerade Zufahrt auf eine enge Kurve zeigt Unterschiede in der Approximationsgenauigkeit zwischen den Polynomen 2. und 3. Grades auf. Während die Länge des Residuenvektors (2-Norm) des Polynom 2. Grades auf bis zu 16 Zentimeter anwächst, bleibt die Länge des Residuenvektors des Polynoms 3. Grades auf maximal 4 Zentimeter begrenzt (vgl. Abbildung 5.7). Die Konsequenz ist die bessere Approximation der Fahrspur des Testszenarios mit dem Polynom 3. Grades (vgl. Bildsequenzen in Anhang F.1 und F.2)

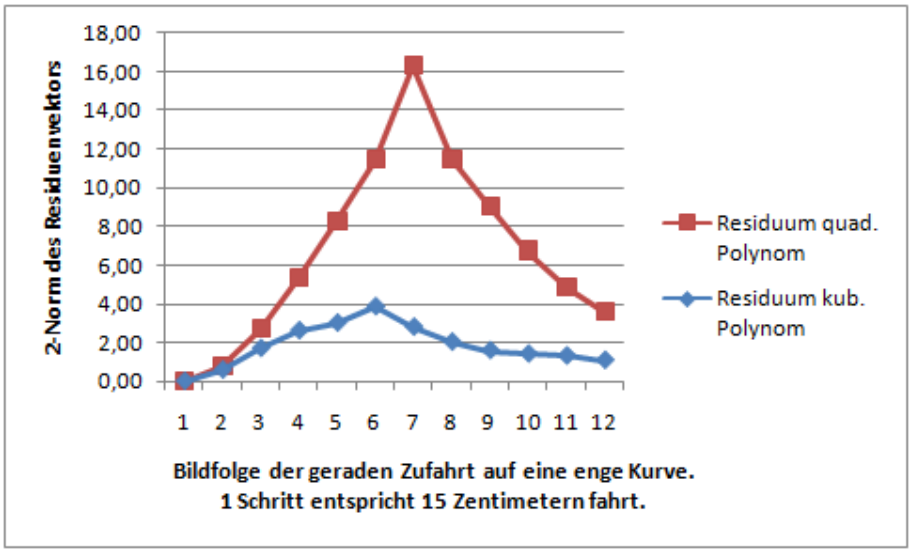


Abb. 5.7: Entwicklung des Residuums bei der Zufahrt auf eine Kurve

Gerade Zufahrt auf eine S-Kurve

Bei der geraden Zufahrt auf eine S-Kurve zeigt sich ebenfalls die größere Beweglichkeit des Polynom 3. Grades gegenüber dem Polynom 2. Grades (vgl. Abbildung 5.8). Zu keinem Zeitpunkt ist die 2-Norm des Residuenvektors vom Polynom 2. Grades geringer als die vom Polynom 3. Grades. Die Bildsequenzen in Anhang F.3 und F.4 weisen die größere Übereinstimmung des Polynom 3. Grades mit den Fahrspurmarkierungspunkten nach.

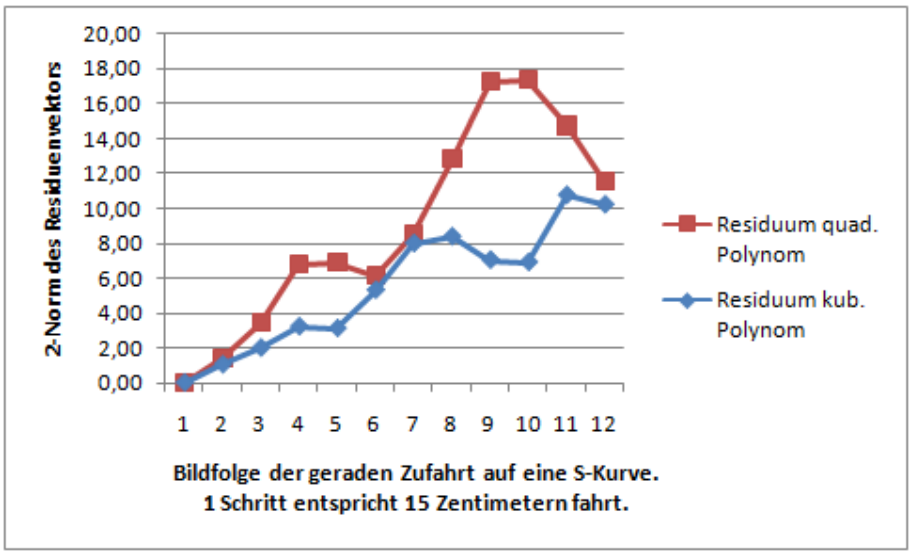


Abb. 5.8: Entwicklung des Residuums bei der Zufahrt auf eine S-Kurve

5.3 Einfluss von Approximationsungenauigkeiten auf die Identifikation

Die in Abschnitt 5.2.4 beschriebenen Approximationsungenauigkeiten wirken sich auf die Identifikation der Fahrzeuglage und des Steuerkurswinkels aus. Nachfolgend werden zunächst die lateralen Abweichungen der Polynome vom vorgegeben Sollwert (100 Zentimeter) bei der geraden Zufahrt auf eine enge Kurve und auf eine S-Kurve betrachtet. Anschließend folgt die Auswertung der Entwicklung des Steuerkurswinkels bei gerader Zufahrt auf eine enge Kurve und auf eine S-Kurve.

Der Schnittpunkt des Polynom 2. Grades mit der y-Achse weicht bei der Zufahrt auf eine enge Kurve um bis zu 5 Zentimeter vom Sollwert ab (vgl. Abbildung 5.9). Der Schnittpunkt des Polynom 3. Grades mit der y-Achse zeigt eine Abweichung von etwa 2 Zentimetern und führt somit zu einem geringeren Fehler in der lateralen Positionsbestimmung.

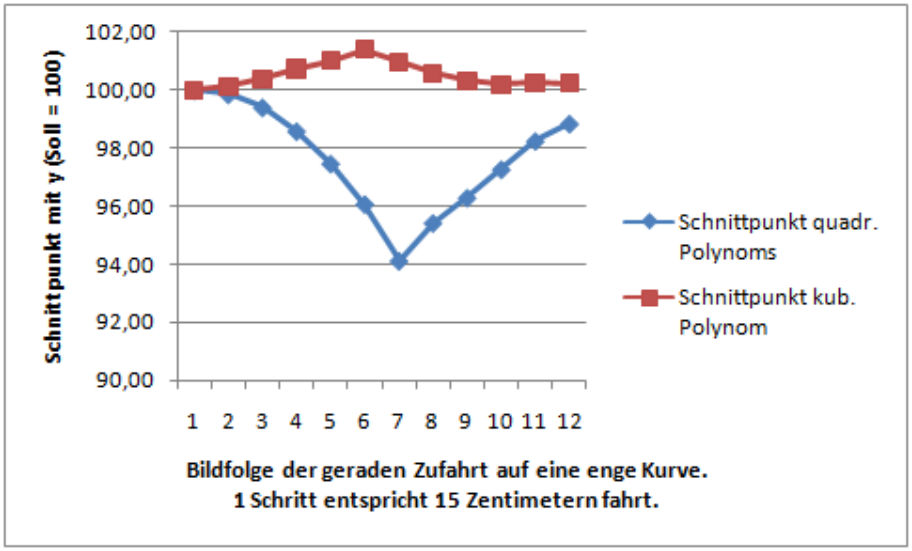


Abb. 5.9: Entwicklung des Schnittpunkts mit der y-Achse bei der Zufahrt auf eine Kurve

Die Zufahrt auf eine S-Kurve zeigt eine Schwankung des Schnittpunkts um den Sollwert für beide Polynome (vgl. Abbildung 5.10). In den Simulationsschritten 7 und 8 zeigt das Polynom 2. Grades eine geringere Abweichung vom Sollwert als das Polynom 3. Grades. In den anderen Fällen stellt das Polynom 3. Grades die genauere Näherung dar. Über die gesamte Bildfolge betrachtet, variieren der Schnittpunkt des Polynoms 2. Grades um etwa 8 Zentimeter und der Schnittpunkt des Polynoms 3. Grades um etwa 4 Zentimeter.

Der Steuerkurswinkel des Fahrzeugs zur Fahrspur wird ebenfalls durch die Genauigkeit der Approximation beeinflusst. Der Sollwert für den Steuerkurswinkel beträgt in diesem Test 0°, da das Fahrzeug ungelenkt entlang einer Geraden fährt. Die Bildsequenzen zur Auswertung des Steuerkurswinkels sind in Anhang F.3 und F.4 zu finden.

Die gerade Zufahrt auf eine enge Kurve zeigt eine Abweichung von maximal 12° für das kubische und -22° für das quadratische Polynom vom Sollwert des Steuerkurswinkels. Auch in diesem Test führt die Approximation durch ein kubisches Polynom gegenüber einem quadratischen Polynom zu geringeren Messfehlern. Die entstehende Abweichung von bis zu 12° sollte

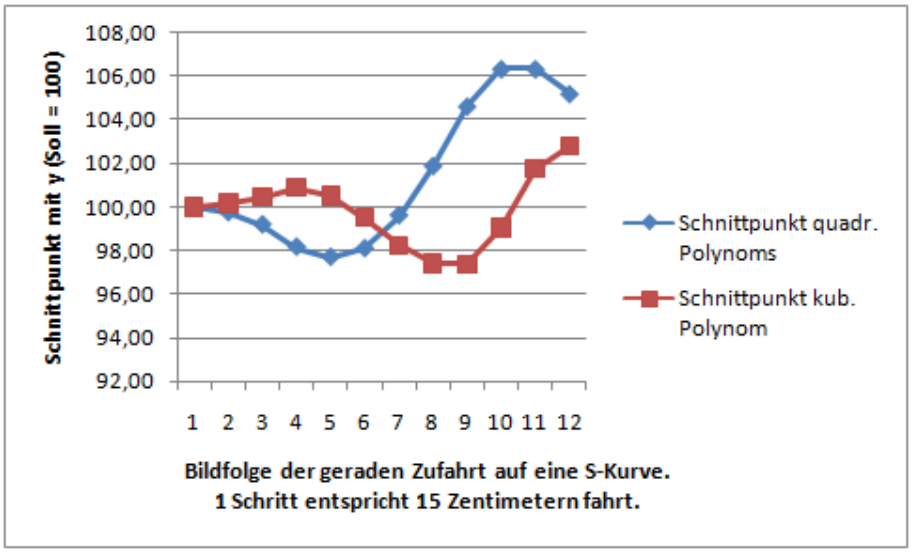


Abb. 5.10: Entwicklung des Schnittpunkts mit der y-Achse bei der Zufahrt auf eine Kurve

bei der Verwendung des Winkels durch eine Filterung über die zeitliche Folge kompensiert werden.

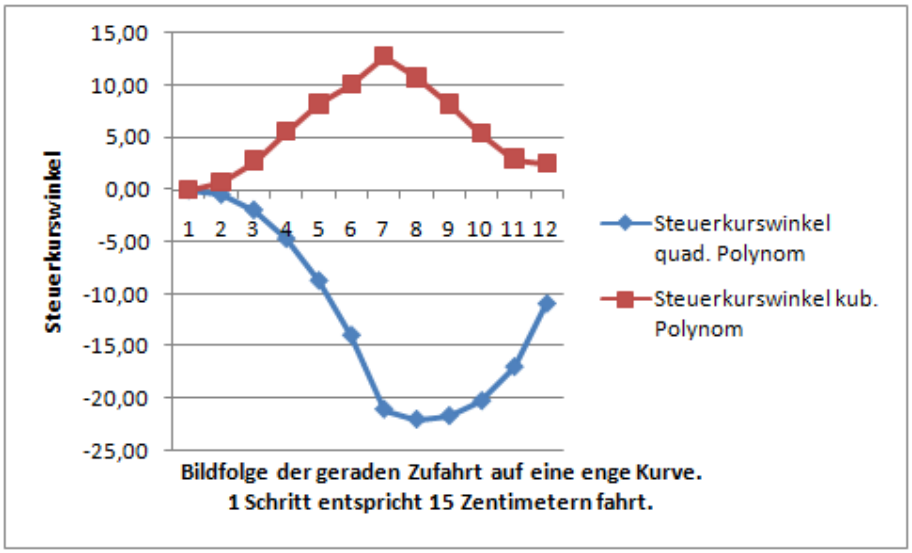


Abb. 5.11: Entwicklung des Steuerkurswinkels bei der Zufahrt auf eine enge Kurve

Die Zufahrt auf eine S-Kurve zeigt größere Abweichung im Steuerkurswinkel als die Zufahrt auf eine enge Kurve. Das quadratische Polynom führt zu Abweichungen von bis zu 25° vom Sollwinkel. Das kubische Polynom weicht um bis zu -20° ab.

Der Steuerkurswinkel zeigt deutlich messbare Abweichungen, sowohl bei der Zufahrt auf eine enge Kurve, als auch bei der Zufahrt auf eine S-Kurve. Aus diesem Grund sollte die Genauigkeit der Approximation weitergehend untersucht und verbessert werden (vgl. Abschnitt 7.1), da die Abweichungen eine Konsequenz der Approximationsungenauigkeit sind.

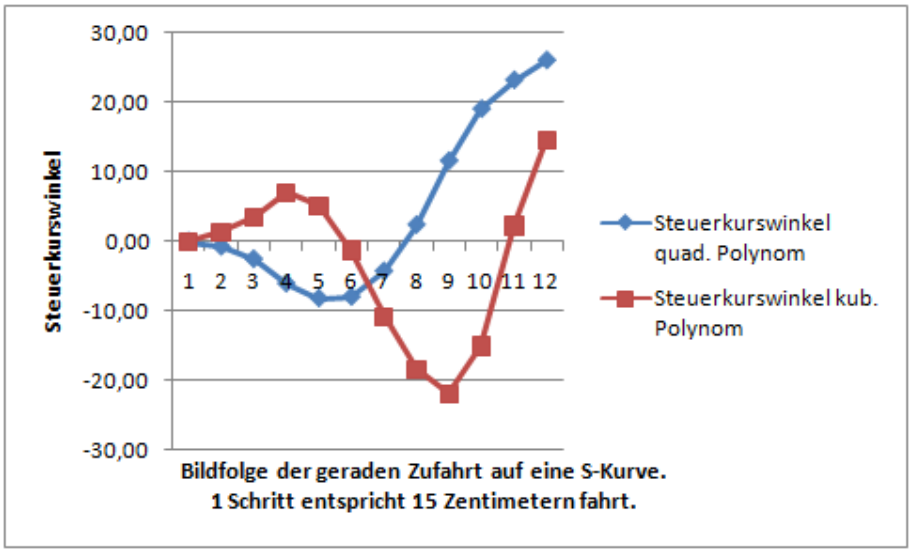


Abb. 5.12: Entwicklung des Steuerkurswinkels bei der Zufahrt auf eine S-Kurve

5.4 Tabellarisch zusammengefasster Zeitaufwand des Verfahrens

Dieser Abschnitt fasst in Form einer Tabelle die maximal gemessenen Laufzeiten der einzelnen Verfahrensschritte auf dem Embedded-PC der Testumgebung (vgl. Tabelle 5.1) zusammen. Die Tabelle ist so aufgebaut, dass das bisherige Fahrspurerkennungsverfahren „TFALDA“ und das in dieser Arbeit entwickelte Fahrspurerkennungsverfahren verglichen werden können. Die Ausführungszeiten werden per Software auf eine Mikrosekunde genau gemessen.

Tabelle 5.10: Maximale Ausführungszeiten der einzelnen Verfahrensschritte auf der eingesetzten Hauptrecheneinheit

Maximale Ausführungszeiten auf dem Via Nano-ITX Embedded-PC

Verfahrensschritt	entw. Verfahren [μs]	„TFALDA“ [μs]
Fahrspurdiskretisierung	1552	26962
Linsenverzeichnungskorrektur	58	32
Projektive Transformation	25	17
Polynomapproximation	131	119
Identifikation	14	14
Gesamtaufwand	$\approx$ 1800	$\approx$ 27100

Die nachfolgenden Grafiken zeigen die gemessenen Zeiten der Verfahrensschritte bezogen auf die ausgewertete Bildsequenz. Auf die Darstellung der Ausführungszeiten der Systemidentifikation wird verzichtet, da dort keine Differenzen zwischen dem in dieser Arbeit entwickelten Verfahren und „TFALDA“ erkennbar sind.

Die Fahrspurerkennung macht den wesentlichen Geschwindigkeitsunterschied zwischen den untersuchten Verfahren aus (vgl. Abbildung 5.13). Dies ist auf die Reduzierung auf einzeilige Betrachtungsbereiche und auf den Verzicht einer lokale Optimierung zurück zu führen. Bei der Linsenverzeichnungskorrektur ist hingegen die Ausführungszeit von „TFALDA“ aufgrund

der geringeren Anzahl zu korrigierender Punkte besser gegenüber dem in dieser Arbeit entwickelten Verfahren (vgl. Abbildung 5.14). Bei der projektiven Transformation der Punkte liegen beide Verfahren etwa gleichauf (vgl. Abbildung 5.15). Eigentlich müsste auch hier „TFALDA“ schneller sein. Vermutlich ist das Ergebnis auf Ungenauigkeiten der Softwaremessung zurück zu führen. Die Ausgleichsrechnung wird ebenfalls wie erwartet durch „TFALDA“ schneller ausgeführt, da lediglich 4 Messwerte pro Seite vorliegen (vgl. Abbildung 5.16). Demgegenüber werden in dem Verfahren dieser Arbeit 13 Messwerte pro Seite zur Ausgleichsrechnung verwendet. Trotzdem liegen die Ausführungszeiten nah beieinander. Dies lässt auf eine Effiziente implementierung der Singulärwertzerlegung in der opencv-Bibliothek [opencv (2006)] schließen.

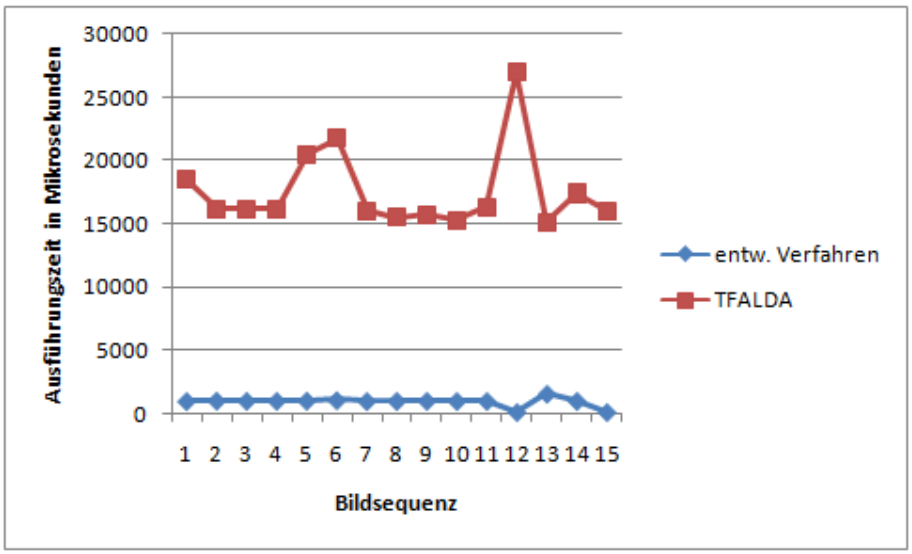


Abb. 5.13: Ausführungszeiten der Fahrspurerkennung in dem Verfahren dieser Arbeit und in TFALDA

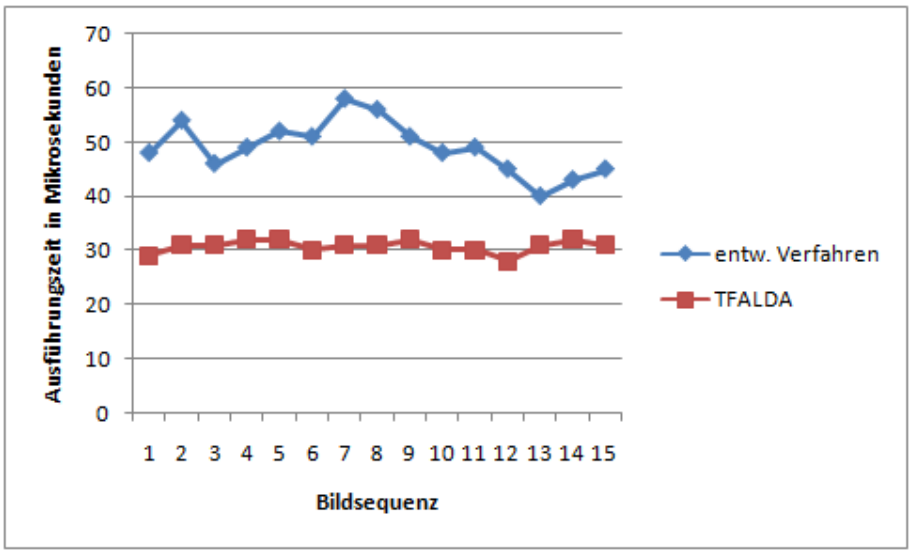


Abb. 5.14: Ausführungszeiten der Linsenverzeichnungskorrektur in dem Verfahren dieser Arbeit und in TFALDA

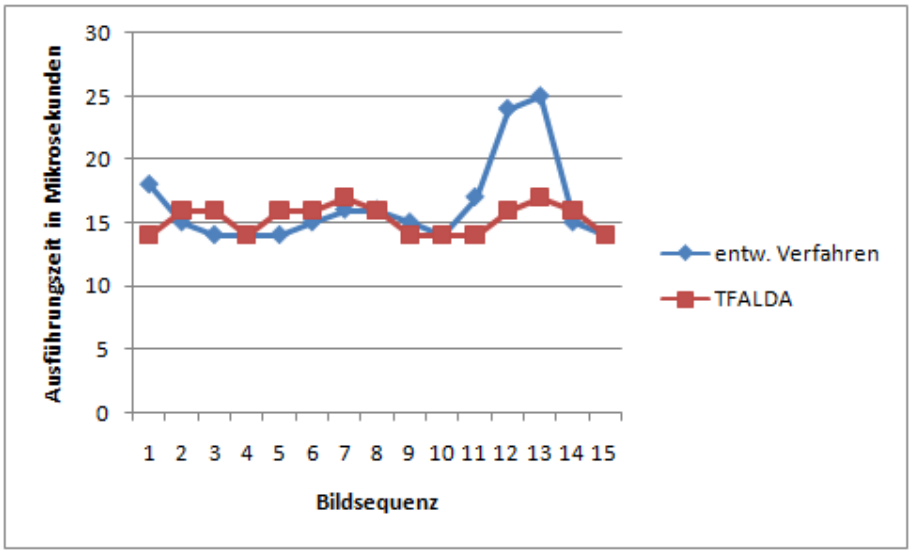


Abb. 5.15: Ausführungszeiten der projektiven Transformation in dem Verfahren dieser Arbeit und in TFALDA

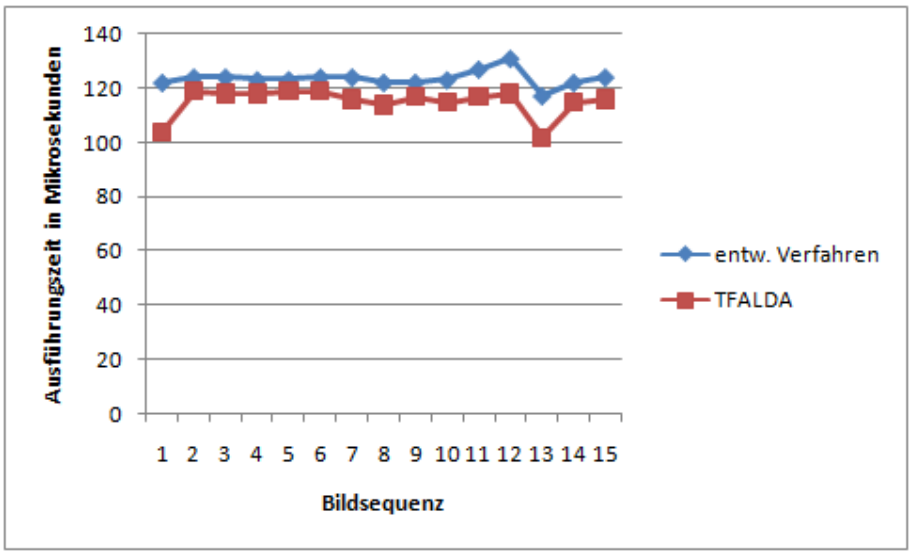


Abb. 5.16: Ausführungszeiten der Ausgleichsrechnung in dem Verfahren dieser Arbeit und in TFALDA

6 Zusammenfassung

Im Rahmen des Forschungsprojekts FAUST (Fahrerassistenz- und Autonome Systeme) wird ein Projekt zur Teilnahme am Carolo-Cup [Carolo-Cup] im Februar 2009 durchgeführt. In dieser Arbeit wurde ein Konzept zur Fahrspurerkennung erarbeitet, um die Realisierung eines neuen Spurführungs-Reglerentwurfs [Enrico Hensel (2008)] und die Umgebungskartierung [Andrej Rull (2008)] systematisch zu unterstützen.

Grundlage des erarbeiteten Konzepts ist ein Polynom 3. Grades (vgl. Abschnitt 2.3), welches zur Approximation von Fahrspurmarkierungen eingesetzt wird. Das Polynom wird in dem, mit dem Fahrzeug bewegten, Fahrzeugkoordinatensystem (vgl. Abschnitt 2.2) berechnet, um eine zentimetergenaue Positionsbestimmung der Fahrspurmarkierung zu realisieren.

Im Bild werden die Fahrspurmarkierungen durch einzelige Betrachtungsbereiche diskretisiert (vgl. Abschnitt 3.1.2). In jedem Betrachtungsbereich wird das Bild mit dem Sobel-Operator gefiltert und anschließend maximal ein Fahrspurmarkierungspunkt extrahiert. Die Betrachtungsbereiche haben einen äquidistanten Abstand im Fahrzeugkoordinatensystem, sodass die Dichte der Betrachtungsbereiche im Bild zum Fluchtpunkt hin zunimmt.

Die Koordinaten der extrahierten Bildpunkte werden durch eine Linsenverzeichnungskorrektur bearbeitet, sodass der Abbildungsfehler des Weitwinkelobjektivs reduziert wird (vgl. Abschnitt 3.2). Die Korrektur erfordert die Kalibrierung der internen Kameraparameter. Es besteht zudem die Option, die Korrektur rückgängig zu machen.

Durch die projektive Transformation werden die verzeichnungsfreien Bildpunkte in das Fahrzeugkoordinatensystem übertragen (vgl. Abschnitt 3.3). Die Berechnung erfordert eine Kalibrierung der Parameter der Projektionsmatrix.

Die Approximation der Parameter des kubischen Polynoms durch die transformierten Punkte erfolgt durch Anwendung der *Methode der kleinsten Quadrate* (vgl. Abschnitt 3.4). Anhand des berechneten Polynoms wird unter Berücksichtigung der aktuellen Fahrzeuggeschwindigkeit und des aktuellen Lenkwinkels die Lage und Orientierung des Polynoms im Folgebild vorhergesagt, um die Betrachtunsbereiche neu auszurichten.

Die Identifikation bestimmt mit dem approximierten Polynom den lateralen Abstand des Fahrzeugs zum Mittelpunkt der Fahrspur, den tangentialen Steuerkurswinkel des Fahrzeugs zur Fahrspur, sowie den Radius an einer beliebigen Stelle des Polynoms (vgl. Abschnitt 4). Diese Parameter werden den Regelungs- und Kartierungssystemen zur Verfügung gestellt.

Die Zielvereinbarungen für die Systemidentifkation sind alle erfüllt (vgl. Abschnitt 1.3). Für die Fahrspurerkennung wurden die Ziele der Echtzeitanforderung und der metrischen Fahrspurdarstellung erreicht. Die Unterstützung für S-Kurven ist gegeben, die messtechnische Auswertung zeigt jedoch Ungenauigkeiten bei der Approximation. Fahrspurwechsel sind dem Konzept noch hinzuzufügen.

7 Ausblick auf weiterführende Entwicklungsschritte

In diesem Kapitel werden konkrete Anregungen über weiterführende Entwicklungsschritte zur Verbesserung und Erweiterung des in dieser Arbeit beschriebenen Verfahrens gegeben.

Zunächst werden notwendige Verbesserungen des Verfahrens in Abschnitt 7.1 genannt. Die Optionen zur Automatisierung bestimmter Verfahrensschritte werden anschließend in Abschnitt 7.2 dargelegt. Eine Erweiterung des Verfahrens um dynamische Komponenten zur Auswertung von Veränderungen über die Zeit wird in Abschnitt 7.3 beschrieben.

7.1 Verbesserung des Verfahrens

Die messtechnische Auswertung zeigt für die projektive Transformation an einer bestimmten Stelle eine Ungenauigkeit, die nicht nachvollzogen werden konnte (vgl. 5.2.3). Die Ursache der Ungenauigkeit sollte für ein vollständiges Verständnis der Projektion systematisch untersucht werden.

Zudem zeigt die messtechnische Auswertung Ungenauigkeiten bei der Approximation einer Fahrspurmarkierung durch ein kubisches Polynom, insbesondere bei S-Kurven auf (vgl. Abschnitt 5.2.4). Dies führt zu einem Fehler bei der Berechnung des lateralen Abstands und des Steuerkurswinkels des Fahrzeugs (vgl. Abschnitt 5.3). Aus diesem Grund sollten weitere Untersuchungen der Approximationseigenschaften durchgeführt werden, um gezielt das Verfahren verbessern zu können.

Für die korrekte Vorhersage der Polynomparameter im Folgebild wird der genaue Lenkwinkel des Modellfahrzeugs benötigt (2.4). Das Fahrzeug sollte deshalb mit einem Sensor zur Lenkwinkelmessung ausgestattet werden.

7.2 Automatisierung von Verfahrensschritten

Für das entwickelte Verfahren wird eine Automatisierung der Kalibrierung der projektiven Transformation (vgl. Abschnitt 3.3) und der initialen Fahrspurerkennung empfohlen.

Zur Kalibrierung der projektiven Transformation wird das Modellfahrzeug vor einem vermessenen Kalibriermuster aufgestellt (vgl. Anhang B) und ein linsenverzeichnungsfreies Bild aufgezeichnet. Anschließend werden manuell die korrespondierenden Bild- und Weltpunkte ermittelt.

Durch eine Automatisierung ließe sich der Aufwand erheblich reduzieren. Zu diesem Zweck könnte eine Bildverarbeitungssoftware im verzeichneten Bild nach den Schnittpunkten des Kalibriermusters suchen, sie der Linsenverzeichnungskorrektur unterziehen und anschließend in Bezug zu den Schnittpunkten in der Welt setzen.

Zur initialen Fahrspurerkennung müssen die Betrachtungsbereiche (vgl. Abschnitt 3.1.2) manuell im Fahrzeugkoordinatensystem angeordnet und die Auswirkung auf die Lage im Bild betrachtet werden. Sobald sich die Startbedingungen des Fahrzeugs verändern, muss die Anordnung angepasst werden. Durch eine Fahrspurerkennung unter Berücksichtigung des gesamten Bildes ließe sich die initiale Anordung der Betrachtungsbereiche automatisieren. So ein Verfahren muss robust gegen schlechte Lichtverhältnisse und große Fahrzeugsteuerkurswinkel sein und zudem Geraden und Kurven erkennen können.

7.3 Erweiterung des Verfahrens um dynamische Komponenten

Das in dieser Arbeit entwickelte Fahrspurerkennung ist ein statisches Verfahren, da das Fahrspurmodell mit jedem Bild neu berechnet wird. Zwar werden die Betrachtungsbereiche durch eine Vorhersage des Polynoms für das Folgebild ausgerichtet, allerdings sind durch Messstörungen immer noch Abweichungen zwischen dem vorhergesagtem und dem ermittelten Polynom möglich (vgl. Abbildung 7.1).

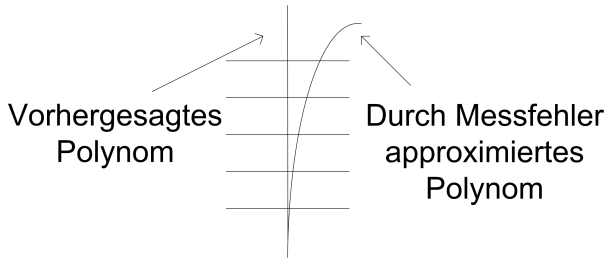


Abb. 7.1: Abweichungen zwischen dem vorhergesagtem und dem approximierten Polynom durch statisches Verfahren

Aus diesem Grund erscheint die Betrachtung des zeitlichen Veränderungsverlaufs der Polynome durch den Aufbau einer Historie sinnvoll. Durch einen Tiefpassfilter können zu große Abweichungen zwischen zwei aufeinander folgenden Polynomen geglättet und konsequenterweise der Erkennungsfehler reduziert werden. Die Größe des Tiefpassfilters sollte mit der gefahrenen Geschwindigkeit variieren, sodass bei einer hohen Geschwindigkeit größere Veränderungen zugelassen werden.

Auch an anderen Stellen des Verfahrens ist eine Tiefpassfilterung über eine Historie wünschenswert. Es lassen sich ähnlich wie bei „TFALDA“ (3.1.1) Bedingungen an den Fahrspurverlauf, beispielsweise Parallelität, stellen und in das Verfahren intergrieren. Dadurch lassen sich Abweichungen zwischen den linken und rechten approximierten Polynomen glätten.

Eine zusätzliche dynamische Komponente kann eine adaptive Anpassung der Betrachtungsbereichsbreite realisieren. Insbesondere durch Fahrzeugneigung ändert sich die Position der

Fahrspur im Bild sprunghaft (vgl. Abschnitt 5.2.2). Zur Adaption werden nach jedem Fahrspurerkennungszyklus die lateralen Veränderungen der Fahrspur erfasst und anhand der Daten die Parameter einer Wahrscheinlichkeitsverteilung bestimmt. Bei der Annahme einer Normalverteilung könnte die Breite beispielsweise variabel 2σ betragen, sodass etwa 95% der bisher erkannten Fahrspurmarkierungen im Betrachtungsbereich liegen. Für ein mechanisch stabiles Fahrzeug mit geringen Neigungswinkeln wäre die Konsequenz eine adaptive Verringerung der Betrachtungsbereichsbreite und somit ein Geschwindigkeitszuwachs des Verfahrens.

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Literaturverzeichnis

[Carolo-Cup] *Homepage des Carolo-Cup Wettbewerbs*. – http://www.carolo-cup.de/

[opencv 2006] *OpenCV Library*. Version 1.0. 2006. – http://opencv.willowgarage.com/wiki/

[Carolo-Cup-Regelwerk 2009] *Carolo-Cup Regelwerk 2009*, Mai 2009. – http://www.carolo-cup.de/uploads/media/20080507_Carolo-Cup_Regelwerk.pdf

[Alexander Kant 2007] ALEXANDER KANT: *Bildverarbeitungsmodul zur Fahrspurerkennung für ein autonomes Fahrzeug*. Bachelorarbeit, Hochschule für Angewandte Wissenschaften. 2007

[Andrej Rull 2008] ANDREJ RULL: *Sensorbasierte Umgebungskartierung mit lokaler Positionskorrektur für autonome Fahrzeuge*. Bachelorarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Axel Gern u. a. 2002] AXEL GERN ; RAINER MOEBUS ; UWE FRANKE: *Vision-based Lane Recognition under Adverse Weather Conditions Using Optical Flow*. 2002. – http://ieeexplore.ieee.org/iel5/8453/26633/01188025.pdf?tp=&arnumber=1188025&isnumber=26633

[Claudio Rosito Jung und Christian Roberto Kelber 2004] CLAUDIO ROSITO JUNG ; CHRISTIAN ROBERTO KELBER: A Robust Linear-Parabolic Model for Lane Following. In: *17th Brazilian Symposium on Computer Graphics and Image Processing*, 2004, S. 72–79. – http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1352945&isnumber=29722

[Deepak Khosla 2002] DEEPAK KHOSLA: Accurate estimation of forward path geometry using two-clothoid road model. In: *IEEE Intelligent Vehicle Symposium* 1 (2002), June, S. 154–159. – http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1187944&isnumber=26631

[Denis Schetler 2007] DENIS SCHETLER: *Automatischer Ausweichassistent mit einer Laserscanner - basierten Abstandsregelung für ein fahrerloses Transportsystem*. Masterarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2007

[Dennis Berger 2008a] DENNIS BERGER: *Fahrspurerkennung mit Three Feature Based Lane Detection Algorithm (TFALDA)*. Studienarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Dennis Berger 2008b] DENNIS BERGER: *Lane Inference System für den Fahrspuralgorithmus Three Feature Based Lane Detection Algorithm (TFALDA)*. Diplomarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Eike Jenning 2008a] EIKE JENNING: *CaroloCup 2008 - Konzepte und Realisierung*. Projekt - Bericht, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Eike Jenning 2008b] EIKE JENNING: *Fahrspurerkennung in Videoechtzeit mit System-on-chip*. Seminar - Bericht, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Enrico Hensel 2008] ENRICO HENSEL: *Führungskonzept eines autonomen Fahrzeuges*. Anwendung 1 - Bericht, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Ernst D. Dickmanns und Birger D. Mysliwetz 1992] ERNST D. DICKMANNS ; BIRGER D. MYSLIWETZ: Recursive 3-D Road and Relative Ego-State Recognition. In: *IEEE Transactions On Intelligent Transportation Systems* 14 (1992), February, Nr. 2, S. 199–213. – http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=121789&isnumber=3469

[Karl Kluge 1994] KARL KLUGE: Extracting Road Curvature and Orientation From Image Edge Points Without Perceptual Grouping Into Features. In: *Proceedings of the Intelligent Vehicles '94 Symposium*, 1994, S. 109–114. – http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=639482&isnumber=13852

[Karl Kluge und Charles Thorpe 1992] KARL KLUGE ; CHARLES THORPE: Representation and Recovery of Road Geometry in YARF. In: *IEEE Proceedings of the Intelligent Vehicles '92 Symposium*, 1992, S. 114–119. – http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=252242&isnumber=6442

[Karl Kluge und Sridhar Lakshmanan 1995] KARL KLUGE ; SRIDHAR LAKSHMANAN: A Deformable-Template Approach to Lane Detection. In: *Proceedings of the Intelligent Vehicles '95 Symposium*, 1995, S. 54–59. – http://ieeexplore.ieee.org/iel3/4019/11537/00528257.pdf?tp=&arnumber=528257&isnumber=11537

[Keith A. Redmill u. a. 2001] KEITH A. REDMILL ; SRINIVASA UPADHYA ; ASHOK KRISHNAMURTHY ; ÜMIT ÖZGÜNER: A Lane Tracking System for Intelligent Vehicle Applications. In: *IEEE Intelligent Transportation Systems* 1 (2001), August, S. 273–279. – http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=948668&isnumber=20514

[Lothar Papula 2007] LOTHAR PAPULA: *Mathematik für Ingenieure und Naturwissenschaftler*. Bd. Band 1. 11., verbesserte und erweiterte Auflage. Vieweg, 2007. – ISBN: 978-3-8348-0224-8

[Martin Hermann 2006] MARTIN HERMANN: *Numerische Mathematik*. 2. Auflage. Oldenbourg Wissenschaftsverlag, 2006. – ISBN: 3-486-57935-5

[Michael Ebert 2008] MICHAEL EBERT: *Aktives Mapping und Positionsbestimmung eines autonomen Fahrzeugs*. Bachelorarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Nico Manske 2008] NICO MANSKE: *Kamerabasierte Präzisionsnavigation mobiler Systeme im Indoor-Bereich*. Masterarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Nico Manske und Thorsten Jost 2008] NICO MANSKE ; THORSTEN JOST: *Posenbestimmung in Räumen mit einem 3-D Kameramodell*. Projekt - Bericht, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Nils Kruse 2008] NILS KRUSE: *Kameragestützte Fahrspurerkennung für autonome Modellfahrzeuge*. Bachelorarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2008

[Norbert Herrmann 2004] NORBERT HERRMANN: *Höhere Mathematik*. Oldenbourg Wissenschaftsverlag, 2004. – ISBN: 3-486-27498-8

[Peter Hartmann 2006] PETER HARTMANN: *Mathematik für Informatiker*. 4., überarbeitete Auflage. Vieweg, 2006. – ISBN: 3-8348-0096-1

[R. Risack u. a. 1998] R. RISACK ; P.KLAUSMANN ; W. KRÜGER ; W. ENKELMANN: Robust lane recognition embedded in a real-time driver assistance system. In: *IEEE International Conference on Intelligent Vehicles* Bd. 1, 1998, S. 35–40

[Richard Hartley und Andrew Zisserman 2003] RICHARD HARTLEY ; ANDREW ZISSERMAN: *Multiple View Geometry in computer vision*. Second Edition. Cambridge University Press, 2003. – ISBN: 0-521-54051-8

[Richard Matthaei 2007] RICHARD MATTHAEI: *Entwurf einer kamerabasierten Fahrspurerkennung für ein autonom fahrendes Modellauto*. Studienarbeit, Institut für Regelungstechnik, Technische Universität Braunschweig. 2007

[Stephan Neumaier und Georg Färber 2005] STEPHAN NEUMAIER ; GEORG FÄRBER: Videobasierte Fahrspurerkennung zur Umfelderfassung bei Straßenfahrzeugen. In: *Autonome Mobile Systeme* Teil 5 (2005), S. 173–178

[Tarik Mouslih 2007] TARIK MOUSLIH: *Visuelle Navigation eines autonomen Fahrzeugs mit Hilfe von Bildverarbeitung*. Diplomarbeit, Hochschule für Angewandte Wissenschaften Hamburg. 2007

[Thomas Huckle und Stefan Schneider 2002] THOMAS HUCKLE ; STEFAN SCHNEIDER: *Numerik für Informatiker*. Springer-Verlag, 2002. – ISBN: 3-540-42387-7

[William H. Press u. a. 2007] WILLIAM H. PRESS ; SAUL A. TEUKOLSKY ; WILLIAM T. VETTERLING ; BRIAN P. FLANNERY: *Numerical Recipes*. Third Edition. Cambridge University Press, 2007. – ISBN: 978-0-521-88407-5

[Young Uk Yim und Se-Young Oh 2003] YOUNG UK YIM ; SE-YOUNG OH: Three-Feature Based Automatic Lane Detection Algorithm (TFALDA) for Autonomous Driving. In: *IEEE Transactions On Intelligent Transportation Systems* 4 (2003), Dezember, Nr. 4, S. 219–225. – http://ieeexplore.ieee.org/iel5/6979/28168/01260588.pdf?tp=&arnumber=1260588&isnumber=28168

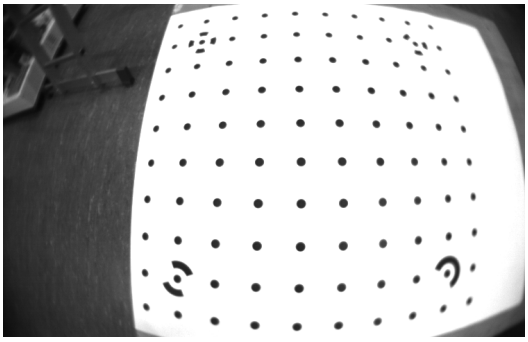
[Zhu Wennan u. a. 2006] ZHU WENNAN ; CHEN QIANG ; WANG HONG: Lane Detection in Some Complex Conditions. In: *Proceedings of the 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006, S. 117–122. – http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4058534&isnumber=4058335

A Kalibrierung der internen Kameraparameter für die Einsatzumgebung

In diesem Kapitel werden die kalibrierten internen Parameter der Kamera der Testumgebung aufgelistet und die Kalibrierungsaufnahmen dargestellt.

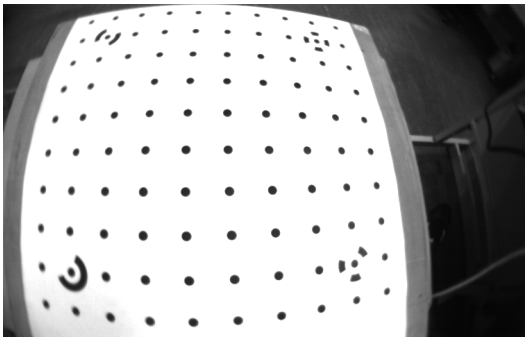
Tabelle A.1: Daten der Plattform-Kamera uEye1226LE-M inkl. interner Parameter

Parameter	Wert
x_{bh} [mm]	3.0238
y_{bh} [mm]	1.8584
Brennweite [mm]	2.8766
K1	0.02975
K2	0.004184
P1	0.0005559
P2	-0.0002571
Chipbreite [mm]	6.0108
Chiphöhe [mm]	3.8298
Auflösung	752 x 480

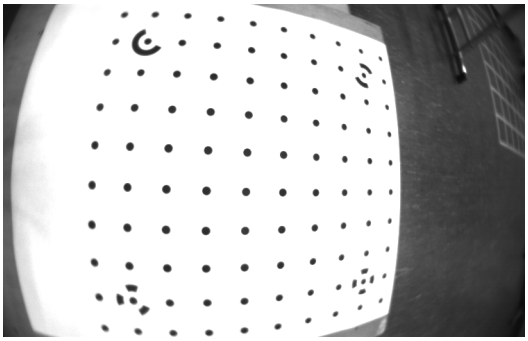


(a) Perspektive 1 Winkel 0°

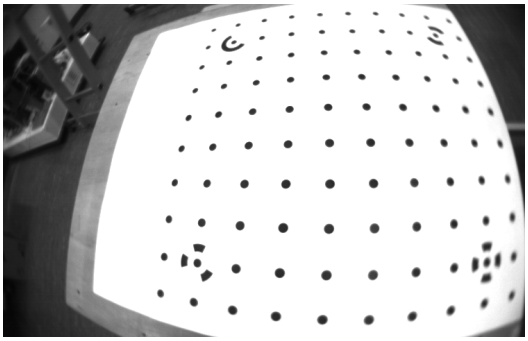
(b) Perspektive 1 Winkel 90°



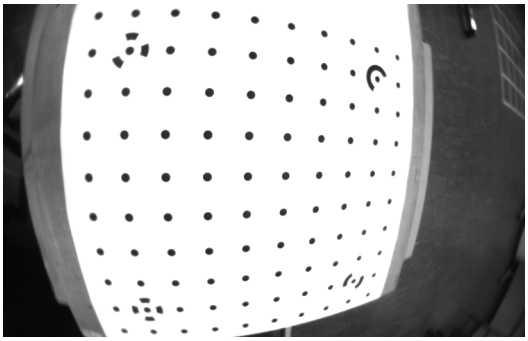
(c) Perspektive 2 Winkel 0°



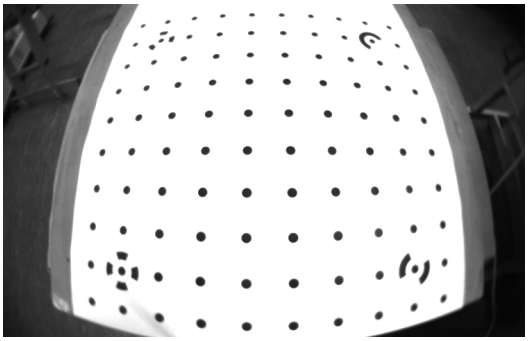
(d) Perspektive 2 Winkel 90°



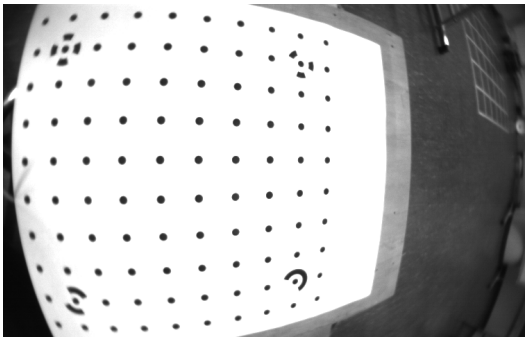
(e) Perspektive 3 Winkel 0°



(f) Perspektive 3 Winkel 90°



(g) Perspektive 4 Winkel 0°

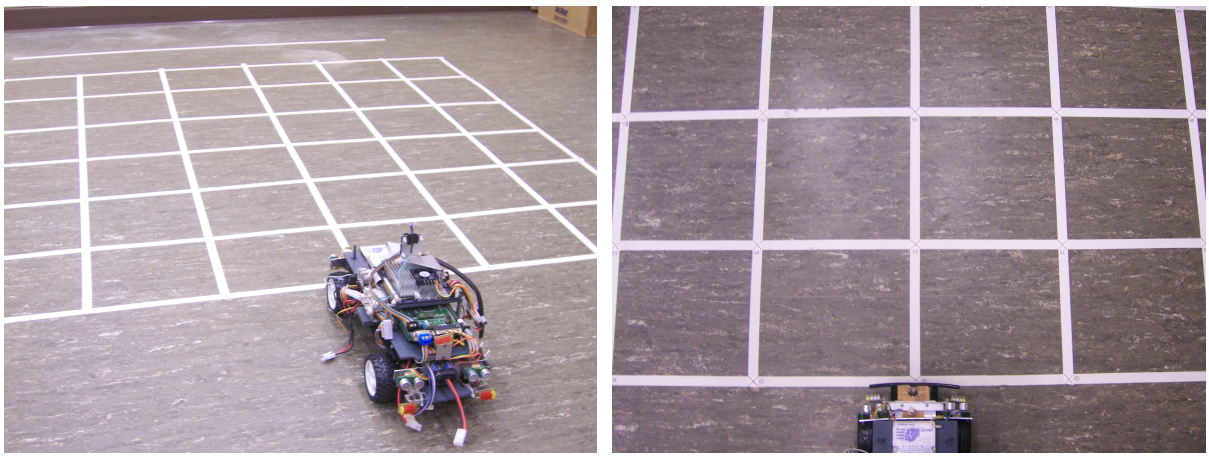


(h) Perspektive 4 Winkel 90°

Abb. A.1: Aufnahmen der Kalibrieranordnung aus verschiedenen Perspektiven und Winkeln

B Kalibrierung der projektiven Transformation für die Einsatzumgebung

In diesem Kapitel werden Aufnahmen von der Kalibrierung des Fahrzeugs dargestellt. Zur Bestimmung der korrespondierenden Punkte werden im linsenverzeichnungskorrigierten Kamerabild (vgl. Abbildung B.2) die Schnittpunkte innerhalb des Kalibriermusters betrachtet und deren Pixelkoordinaten mit den vermessenen Weltkoordinaten zusammengetragen (vgl. Abbildung B.3).



(a) Aufbau des Fahrzeugs vor dem Kalibriermuster der projektiven Transformation, Sicht von hinten

(b) Aufbau des Fahrzeugs vor dem Kalibriermuster der projektiven Transformation, Sicht von oben

Abb. B.1: Aufstellen des Fahrzeugs im Ursprung des Kalibriermusters der projektiven Transformation

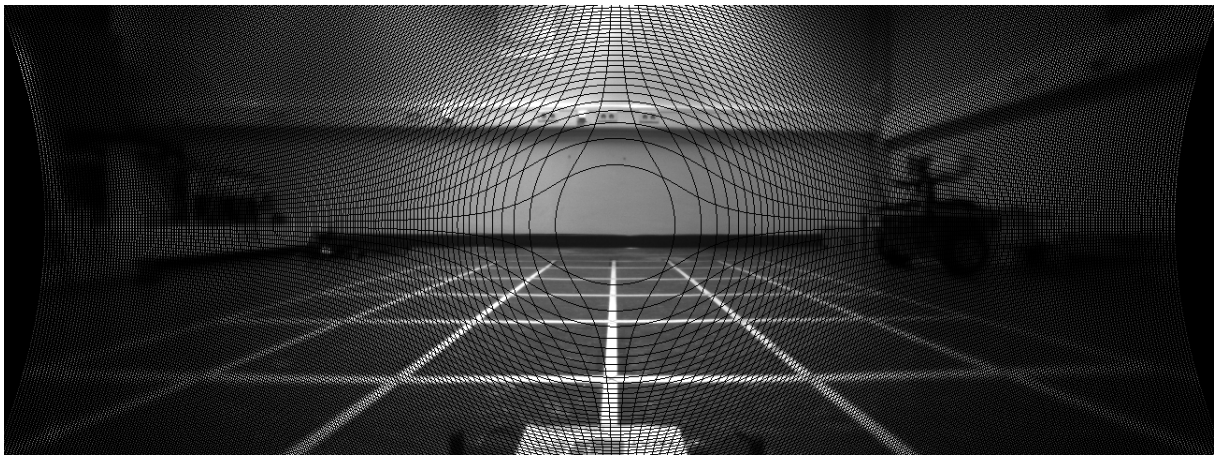


Abb. B.2: Linsenverzeichnungsfreie Aufnahme des Kalibriermusters mit dem Testfahrzeug

X in cm

ID 7 x = 160.072232 y = -89.873572 x* = 195 y* = 273	ID 6 x = 160.574576 y = -62.284529 x* = 250 y* = 273	ID 5 x = 161.01565 y = -31.970526 x* = 310 y* = 273	ID 4 x = 161.446281 y = 0 x* = 374 y* = 273	ID 3 x = 161.60531 y = 29.025286 x* = 434 y* = 273	ID 2 x = 162.050644 y = 56.618562 x* = 487 y* = 272	ID 1 x = 162.505957 y = 81.423051 x* = 535 y* = 272
ID 14 x = 134.12735 y = -89.249922 x* = 169 y* = 279	ID 13 x = 134.501267 y = -61.503748 x* = 231 y* = 280	ID 12 x = 134.775182 y = -31.509933 x* = 301 y* = 280	ID 11 x = 134.799851 y = -0.02541 x* = 374 y* = 280	ID 10 x = 135.22762 y = 29.068146 x* = 441 y* = 280	ID 9 x = 135.486825 y = 56.856479 x* = 502 y* = 279	ID 8 x = 135.806704 y = 81.587061 x* = 556 y* = 279
ID 21 x = 104.718559 y = -88.684054 x* = 127 y* = 291	ID 20 x = 104.99249 y = -60.590022 x* = 204 y* = 291	ID 19 x = 105.295747 y = -31.224104 x* = 286 y* = 291	ID 18 x = 105.369612 y = -0.063812 x* = 374 y* = 292	ID 17 x = 105.452563 y = 29.089156 x* = 457 y* = 292	ID 16 x = 105.692289 y = 57.175619 x* = 535 y* = 291	ID 15 x = 105.738467 y = 82.128135 x* = 599 y* = 291
ID 28 x = 78.009282 y = -87.910125 x* = 63 y* = 307	ID 27 x = 78.003303 y = -59.714512 x* = 160 y* = 308	ID 26 x = 78.202388 y = -30.883338 x* = 264 y* = 308	ID 25 x = 78.367889 y = -0.109082 x* = 374 y* = 308	ID 24 x = 78.40276 y = 28.720634 x* = 477 y* = 308	ID 23 x = 78.255397 y = 57.161963 x* = 575 y* = 307	ID 22 x = 78.050687 y = 82.23987 x* = 661 y* = 307
ID 35 x = 50.70475 y = -87.289478 x* = -43 y* = 335	ID 34 x = 51.311966 y = -58.906896 x* = 118 y* = 336	ID 33 x = 51.8798 y = -30.557641 x* = 225 y* = 336	ID 32 x = 52.231608 y = -0.105492 x* = 373 y* = 336	ID 31 x = 52.609597 y = 28.621437 x* = 510 y* = 335	ID 30 x = 52.928765 y = 57.195947 x* = 641 y* = 333	ID 29 x = 53.335634 y = 82.324226 x* = 758 y* = 332
ID 42 x = 24.718022 y = -86.371844 x* = -257 y* = 392	ID 41 x = 24.854351 y = -58.026312 x* = -66 y* = 395	ID 40 x = 25.048687 y = -30.130656 x* = 144 y* = 398	ID 39 x = 25.224505 y = -0.00025 x* = 372 y* = 398	ID 38 x = 25.421581 y = 28.300961 x* = 585 y* = 395	ID 37 x = 25.73912 y = 57.346823 x* = 782 y* = 392	ID 36 x = 25.797792 y = 82.535075 x* = 945 y* = 383
ID 49 x = -0.606947 y = -85.516758 x* = - y* = -	ID 48 x = -0.461065 y = -57.45587 x* = - y* = -	ID 47 x = -0.180998 y = -29.824556 x* = - y* = -	ID 46 x = 0 y = 0 x* = - y* = -	ID 45 x = 0.025829 y = 28.432897 x* = - y* = -	ID 44 x = 0.296645 y = 57.292932 x* = - y* = -	ID 43 x = 0.249152 y = 82.264779 x* = - y* = -

Y in cm

Abb. B.3: Kalibriermuster der projektiven Transformation für das Testfahrzeug

C Architektur des FAUSTcore Softwaresystems

Der FAUSTcore stellt dem Anwender eine vollständige Ausführungsumgebung seiner definierten Module zur Verfügung. Zudem ermöglicht der FAUSTcore die bequeme Steuerung der Anwendung über eine Web-Oberfläche (vgl. Abbildung C.1). Nachfolgend werden die in dieser Arbeit zur Erstellung der Testsoftware verwendeten Komponenten des FAUSTcore präsentiert (vgl. Abschnitt C.1) und ihre Verwendung aufgezeigt (vgl. Abschnitt C.2).

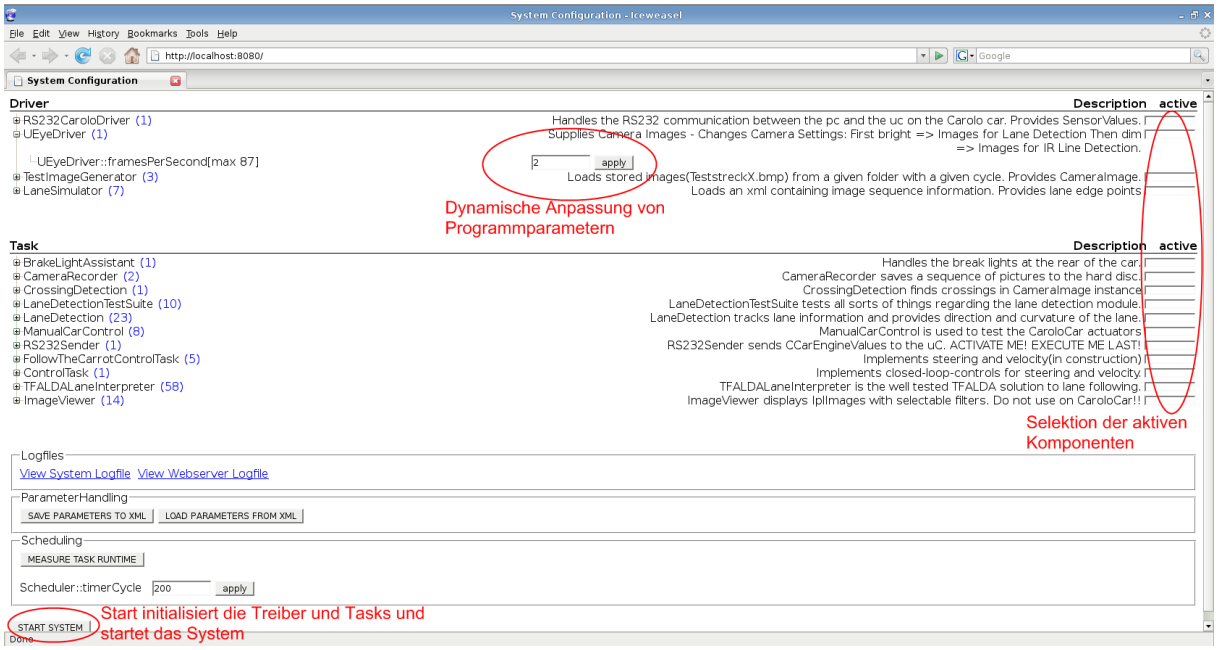


Abb. C.1: Screenshot der Web-Oberfläche des FAUSTcore

C.1 UML-Klassendiagramme der Softwaremodule

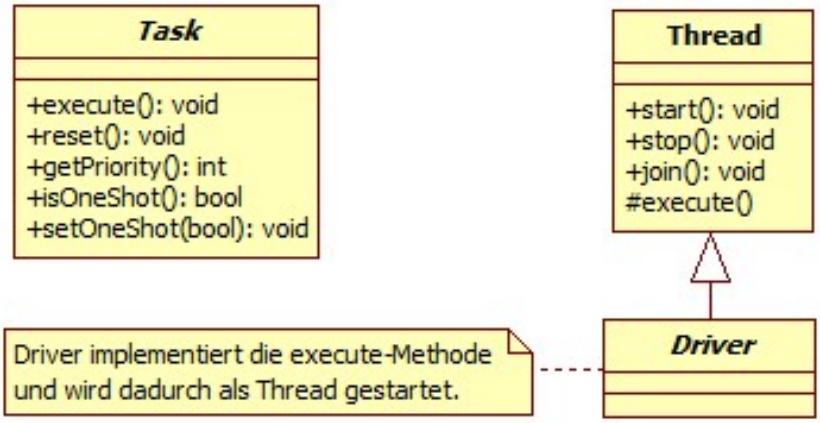


Abb. C.2: Die Abstrakten Klassen Task und Driver von denen der Anwender seine Softwaremodule zur Einbindung in den FAUSTcore ableitet

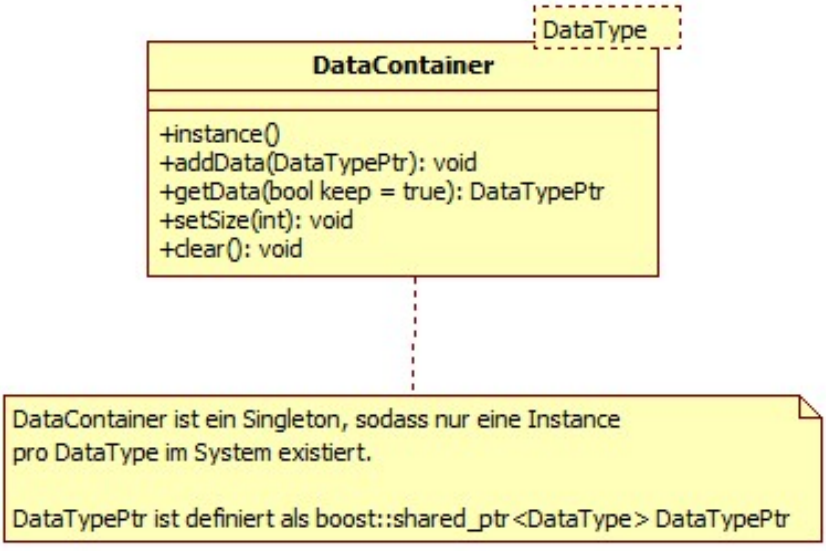


Abb. C.3: Die Klasse DataContainer bietet den systemweiten Zugriff auf bestimmte Datentypen. Typischerweise werden die Daten von einem Driver gespeichert und von einer Task ausgelesen

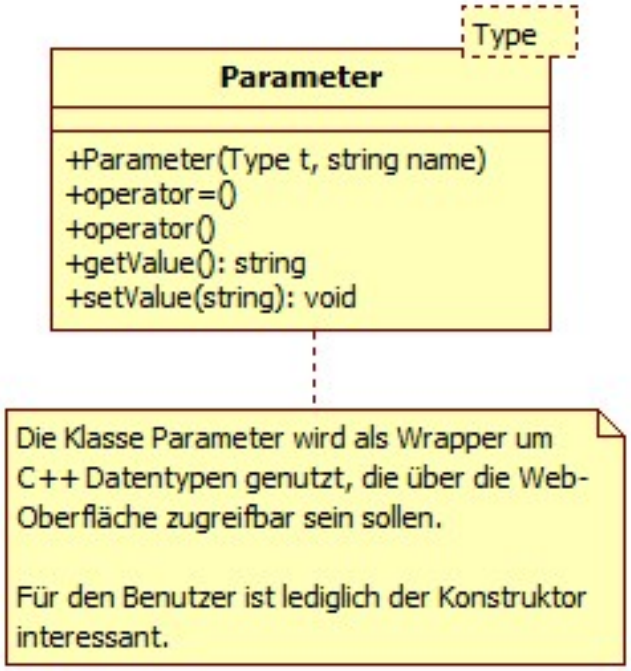


Abb. C.4: Die Klasse Parameter bietet einen Wrapper um C++ Datentypen. Parameter können wie diese verwendet werden und erlauben den Wertezugriff über eine Weboberfläche

C.2 Beispielcode zur Anwendung der Softwaremodule

Listing C.1: Anwendung der FAUSTcore Softwaremodule am Beispiel einer Task LaneDetection

```
1
2#include <Task.h>
3#include <Parameter.h>
4
5class LaneDetection : public Task
6{
7public:
8  LaneDetection();
9  virtual ~LaneDetection();
10
11  // from task
12  virtual void execute();
13
14  // from task
15  virtual void reset();
16
17  // from task
18  virtual bool isOneShot() const
19  {
20    return oneShotParam;
21  }
22
23  // from task
24  virtual int getPriority() const;
25
26private:
27  Parameter<int> priority;
28  Parameter<bool> oneShotParam;
29};
30
31------
32#include "LaneDetection.h"
33
34// these two are needed for automagic loading into FAUSTcore
35// see below
36#include <ObjectFactory.h>
37#include <log4cxx/logger.h>
38
39// for data access
40#include <DataContainer.h>
41
42// initialize the parameters we have
43// in order to access them with your browser you need to put
    your class name
44// followed by :: at the beginning of the parameter name
45LaneDetection::LaneDetection():
```

```
46  priority(0,"LaneDetection::priority"), // highest priority...
47  oneShotParam(true,"LaneDetection::oneShotSetting")
48 {
49 }
50
51 LaneDetection::~LaneDetection()
52 {
53 }
54
55 // calculate scanline positioning in reset method
56 void LaneDetection::reset()
57 {
58   // do something...
59 }
60
61 void LaneDetection::execute()
62 {
63   // receive a camera image from the DataContainer
64   // CameraImagePtr is a boost::shared_ptr<CameraImage>
65   CameraImagePtr image(DataContainer<CameraImage>::instance().
       getData());
66
67   // check if the data is empty
68   if (!image)
69     return;
70
71   // do something...
72 }
73
74 int LaneDetection::getPriority() const
75 {
76   // return the priority parameter as if it was an integer
77   return priority;
78 }
79
80 // this code automagically loads the LaneDetection Task into
     the FAUSTcore
81 // you need to provide it in the cpp file of your class
82 // make sure to change all the names in that snippet according
     to your class name!
83 extern "C" {
84   // the creator
85   Task* createLaneDetection(){
86     return new LaneDetection();
87   }
88   // the proxy to register the creator
89   class LaneDetectionProxy{
90   public:
91     LaneDetectionProxy()
92     {
```

```
93        if (ObjectFactory<Task>::instance().registerCreator(
94            "LaneDetection",
95            createLaneDetection,
96            "Task",
97            "LaneDetection tracks lane information and provides
                direction and curvature of the lane.") == false)
98        {
99          log4cxx::LoggerPtr logger(log4cxx::Logger::getLogger("
               LaneDetection"));
100         LOG4CXX_DEBUG(logger,"unable to register LaneDetection"
               );
101       }
102     }
103   };
104   // create one instance of the proxy
105   LaneDetectionProxy pLaneDetection;
106 }
```

D Softwaremodule des Prototyps zur messtechnischen Auswertung

In diesem Kapitel wird die softwaretechnische Umsetzung der beschriebenen Verfahrensschritte der Approximation und Identifikation als Prototyp skizziert. Nachfolgend werden die einzelnen Softwaremodule beschrieben (vgl. Abschnitt D.1) und anschließend ihre Verwendung aufgezeigt (vgl. Abschnitt D.2).

D.1 UML-Klassendiagramme der Softwaremodule

Die Reihenfolge orientiert sich an dem Ablauf der Verfahrensschritte (vgl. Abbildung 1.1).

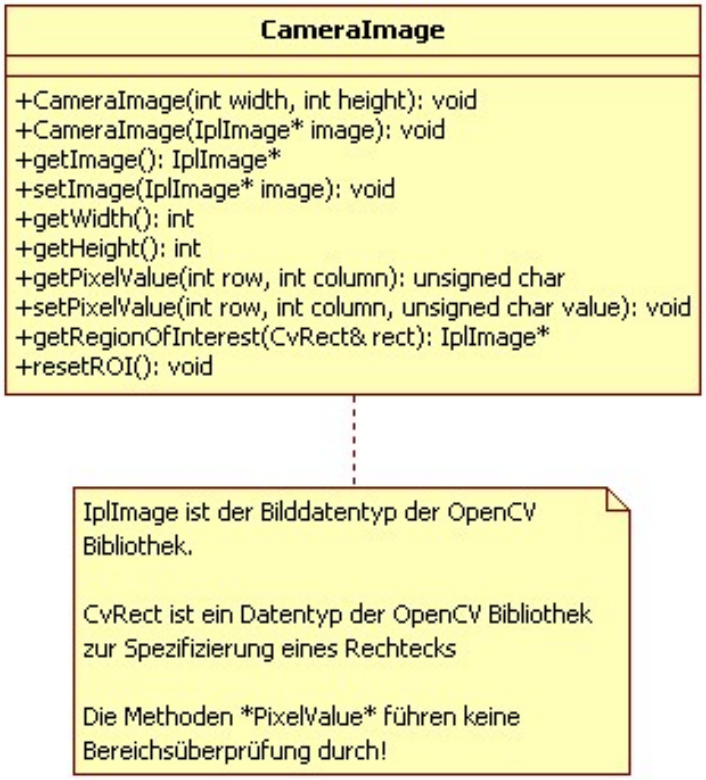


Abb. D.1: Die Klasse CameraImage ist der Bilddatentyp im Softwaresystem. Unter anderem bietet er Zugriff auf einzelne Pixel und die Unterstützung für einen Betrachtungsbereich (ROI)

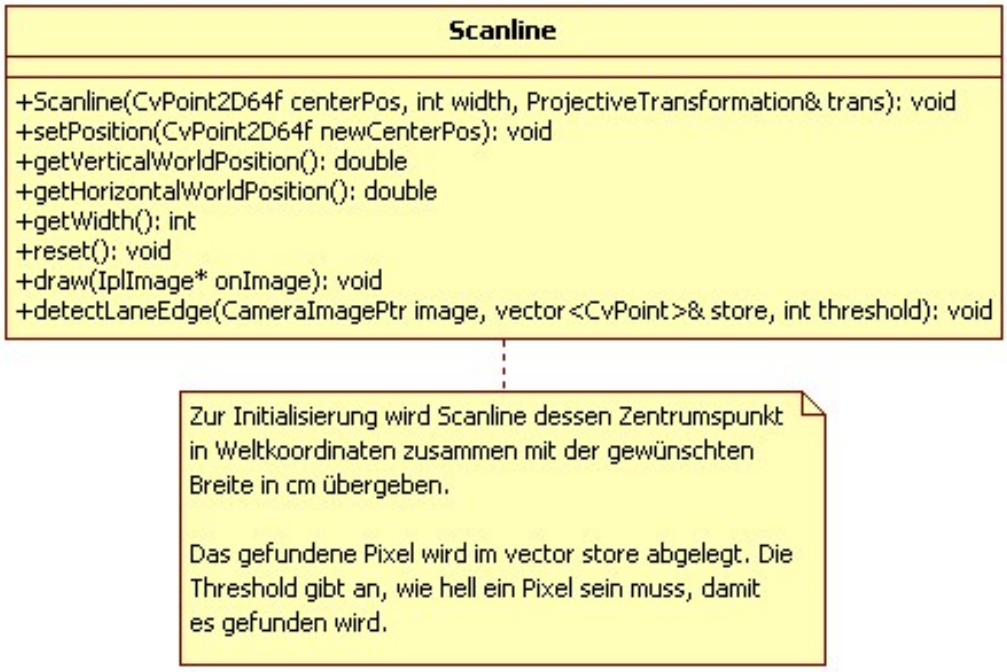


Abb. D.2: Die Klasse Scanline ist die Implementierung eines Betrachtungsbereichs (vgl. Abschnitt 3.1.2)

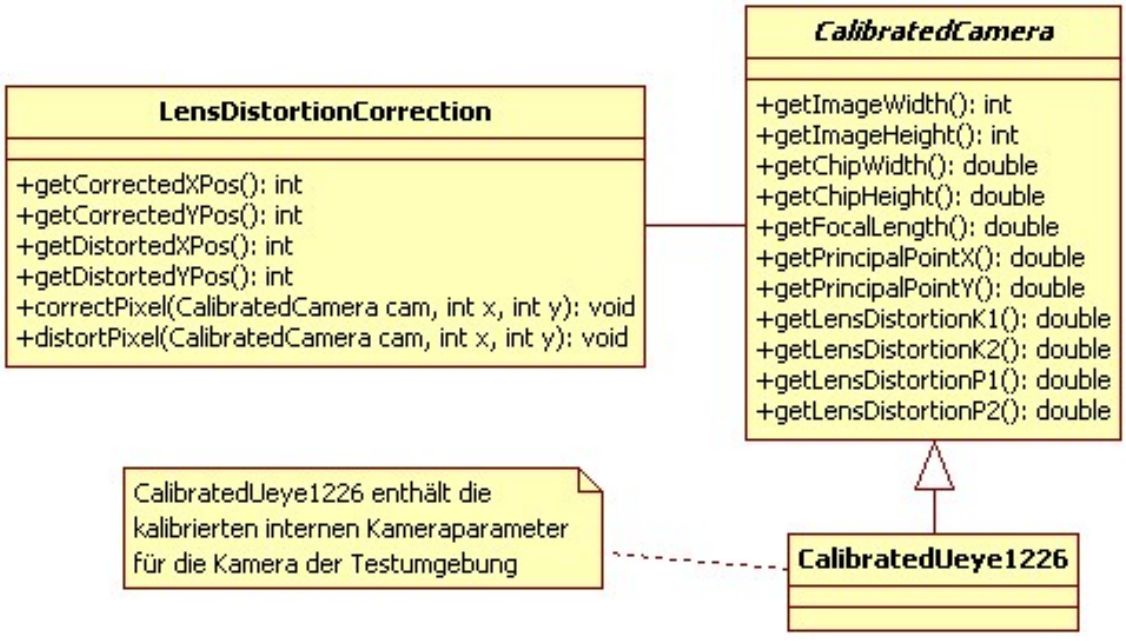


Abb. D.3: Die Klasse LensDistortionCorrection implementiert die Linsenverzeichnungskorrektur (vgl. Abschnitt 3.2). Sie erlaubt sowohl die Korrektur eines verzeichneten Pixels, als auch die Verzeichnung eines korrigierten Pixels.

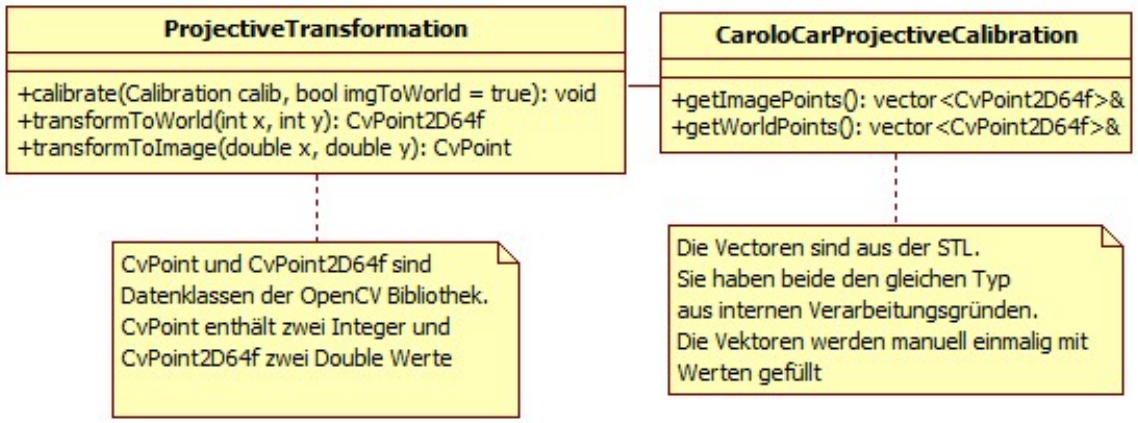


Abb. D.4: Die Klasse ProjectiveTransformation implementiert die projektive Transformation (vgl. Abschnitt 3.3) vom Bild in das Fahrzeugkoordinatensystem und umgekehrt. Vor der Verwendung muss die ProjectiveTransformation mit einer Kalibrierklasse kalibriert werden.

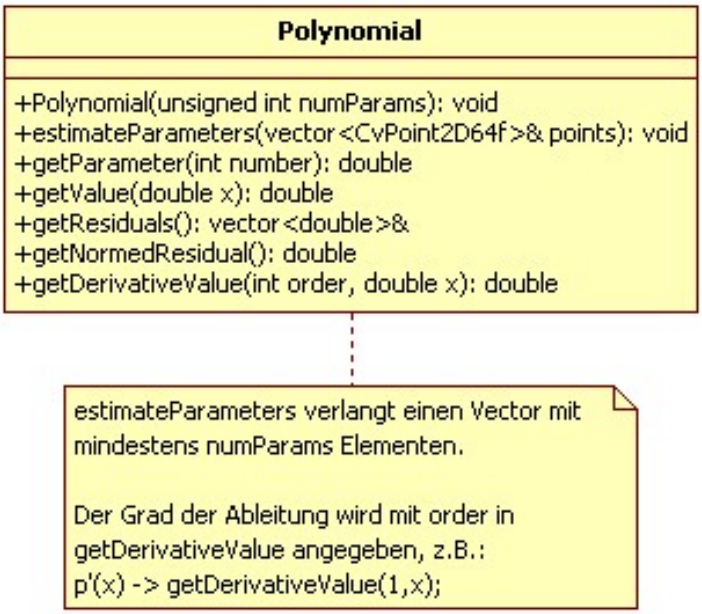


Abb. D.5: Die Klasse Polynomial implementiert das Fahrspurmodell. Sie ist zu Testzwecken mit einer variablen Anzahl an Parametern initialisierbar.

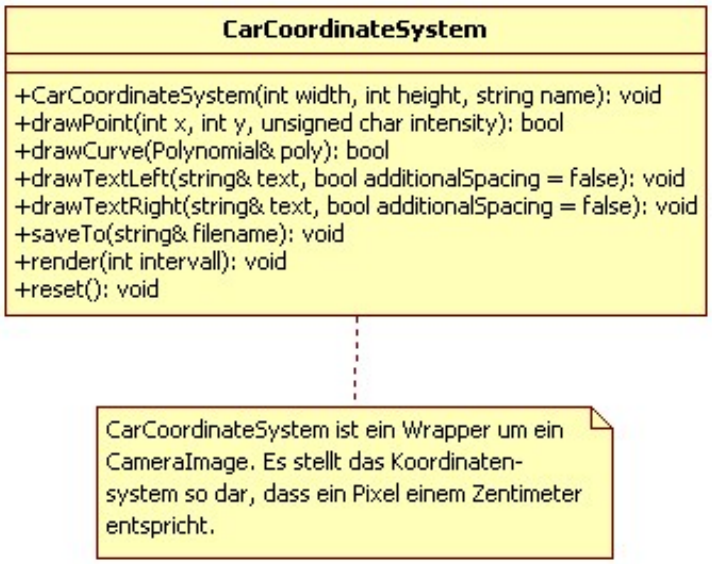


Abb. D.6: Das CarCoordinateSystem implementiert eine bildliche Darstellung des Fahrzeugkoordinatensystems. Mit dieser Hilfsklasse kann das Ergebnis nach der projektiven Transformation und der Ausgleichsrechnung betrachtet werden.

D.2 Beispielcode zur Anwendung der Softwaremodule

Durch die Kapselung der Verfahrensschritte in Module ist eine beliebige Kombiniation zu Testzwecken möglich. Auf diese Weise lassen sich die Ungenauigkeiten der einzelnen Verfahrensschritte bei Bedarf analysieren.

Listing D.1: Beispielcode zur Korrektur der Linsenverzeichnung für ein ganzes Bild

```
1  LensDistortionCorrection correction;
2  CameraImagePtr sourceImg;
3  CameraImagePtr targetImg;
4  CalibratedUeye1226 camera;
5
6  // variables to cache the correction result
7  int correctedCol = 0;
8  int correctedRow = 0;
9
10 // loop through the input image
11 for (int col = 0; col < sourceImg->getWidth(); col++)
12 {
13   for (int row = 0; row < sourceImg->getHeight(); row++)
14   {
15     // correct the pixel coordinate
16     correction.correctPixel(camera,col,row);
17
18     correctedCol = correction.getCorrectedXPos();
19     correctedRow = correction.getCorrectedYPos();
20
21     // check boundaries
22     if ((correctedCol < 0) || (correctedCol > targetImg->
          getWidth()))
23       continue;
24
25     if ((correctedRow < 0) || (correctedRow > targetImg->
          getHeight()))
26       continue;
27
28     // boundaries ok
29     // copy the pixel value from the distorted coordinates to
           the corrected coordinates
30     targetImg->setPixelValue(correctedRow,correctedCol,
          sourceImg->getPixelValue(row,col));
31   }
32 }
33
34 // save targetImg...
```

Listing D.2: Beispielcode zur Kalibrierung der projektiven Transformation

```
// the calibration class that contains the correspondences
CaroloCarProjectiveCalibration ccarProjCal;

// the projective transformation class
ProjectiveTransformation projTrans;

// this calibrates the image to world direction
projTrans.calibrate<CaroloCarProjectiveCalibration>(
   ccarProjCal);

// this calibrates the world to image direction
projTrans.calibrate<CaroloCarProjectiveCalibration>(
   ccarProjCal,false);
```

Listing D.3: Beispielcode zur Anwendung projektiven Transformation

```
// use a projective transformation
ProjectiveTransformation projTrans; // needs calibration...

// calculate the position of several test points
std::vector<CvPoint> points;
// fill image points...

std::vector<CvPoint2D64f> references;
// fill reference points...

for(unsigned int i = 0; i < points.size(); i++)
{
  int x = points[i].x;
  int y = points[i].y;

  CvPoint2D64f wPoint = projTrans.transformToWorld(x,y);
  CvPoint2D64f refPoint = references[i];

  LOG4CXX_DEBUG(logger,"::execute(): difference is ["<< std::
     fabs(refPoint.x) - std::fabs(wPoint.x)
                          << ", "
                          << std::fabs(refPoint.y) - std::
                             fabs(wPoint.y)
                          << "]");
}
```

Listing D.4: Beispielcode zum Einzeichnen eines Polynoms in das Fahrzeugkoordinatensystems

```
// create a polynomial with a number
// of parameters
Polynomial poly(numParams);

// the points to fit the polynomial into
// these are always points in the car coordinate system
std::vector<CvPoint2D64f> points;
// add points here...

// fit the polynomial
poly.estimateParameters(points);

// create a CarCoordinateSystem
// width = 400 cm, height = 200 cm
CarCoordinateSystem ccSystem(400,200,"MyCCSystem");

// draw the polynomial into the CarCoordinateSystem
// pixel intensity = 100
ccSystem.drawCurve(poly,100);

// draw the points as well
for (unsigned int i = 0; i < points.size(); i++)
{
  ccSystem.drawPoint(static_cast<int>(points[i].x),
     static_cast<int>(points[i].y),255);
}

// save and show the ccSystem
ccSystem.saveTo("myfile.png");
ccSystem.render(10); // 10 ms for event processing in gtk
```

E Partielle Ableitungen zur Linsenverzeichnungskorrektur

Nachfolgend werden die partiellen Ableitungen zu den Funktionen

$$f_1(x_m, y_m) = x_m \cdot (1 + K_1 \cdot r^2 + K_2 \cdot r^4) + P_1 \cdot (r^2 + 2 \cdot x_m^2) + 2 \cdot P_2 \cdot x_m \cdot y_m - x_c \tag{E.1}$$

$$f_2(x_m, y_m) = y_m \cdot (1 + K_1 \cdot r^2 + K_2 \cdot r^4) + P_2 \cdot (r^2 + 2 \cdot y_m^2) + 2 \cdot P_1 \cdot x_m \cdot y_m - y_c \tag{E.2}$$

aus Abschnitt 3.2 gebildet.

Nach der Umformung ergibt sich für die partiellen Ableitungen von f_1:

$$\frac{\partial f_1}{\partial x_m} = 1 + K_1 * 3x_m^2 + K_2 * (5x_m^4 + 6x_m^2 y_m^2) + P_1 * 6x_m + P_2 * 2y_m \tag{E.3}$$

$$\frac{\partial f_1}{\partial y_m} = K_1 * 2x_m y_m + K_2 * (4x_m^3 y_m + 4x_m y_m^3) + P_1 * 2y_m + P_2 * 2x_m \tag{E.4}$$

Die partiellen Ableitungen von f_2 berechnen sich zu:

$$\frac{\partial f_2}{\partial x_m} = K_1 * 2x_m y_m + K_2 * (4x_m^3 y_m + 4x_m y_m^3) + P_2 * 2x_m + P_1 * 2y_m \tag{E.5}$$

$$\frac{\partial f_2}{\partial y_m} = 1 + K_1 * 3y_m^2 + K_2 * (6x_m^2 y_m^2 + 5y_m^4) + P_2 * 6y_m + P_1 * 2x_m \tag{E.6}$$

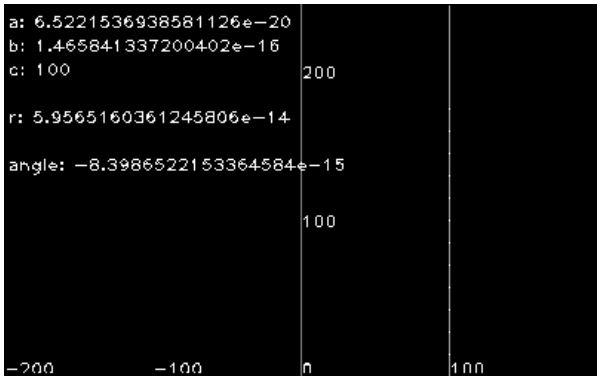
Die partiellen Ableitungen werden zur Lösung des Gleichungssystems mit dem Newton-Verfahren in die Jakobimatrix eingesetzt (vgl. Gleichung 3.15).

F Bildsequenzen der Auswertung der Polynomapproximation

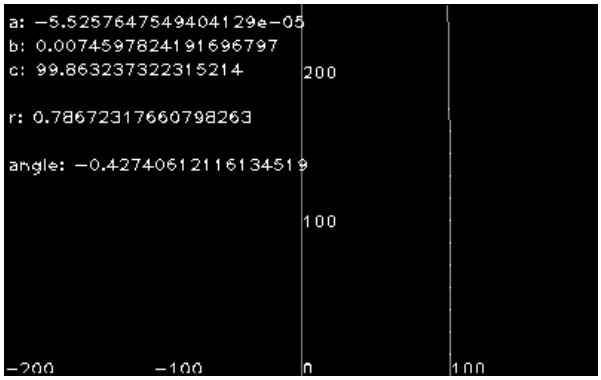
Die Auswertung der quadratischen und kubischen Polynome erfolgt durch eine Simulation von Fahrspurmarkierungspunkten im Fahrzeugkoordinatensystem (vgl. Abschnitt 5.2.4). Nachfolgend werden die zur Auswertung generierten Bildsequenzen dargestellt.

Die Bildsequenz stellt das Fahrzeugkoordinatensystem mit dem Fahrzeug im Ursprung dar. Die x-Achse verläuft als vertikale Linie in der Mitte jedes Bildes. Auf der linken Seite sind die Polynomparameter gefolgt von der 2-Norm des Residuenvektors (vgl. Abschnitt 3.4.1) eingezeichnet.

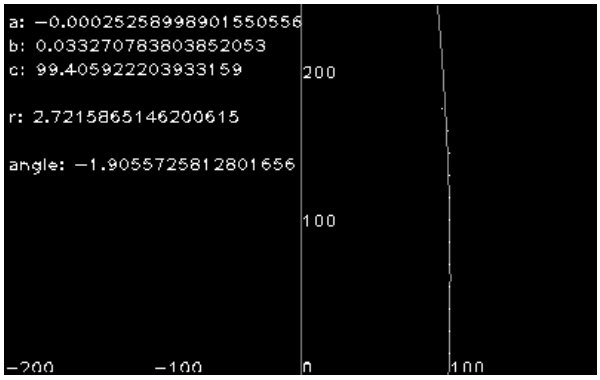
F.1 Bildsequenz des Polynom 2. Grades bei Zufahrt auf eine enge Kurve



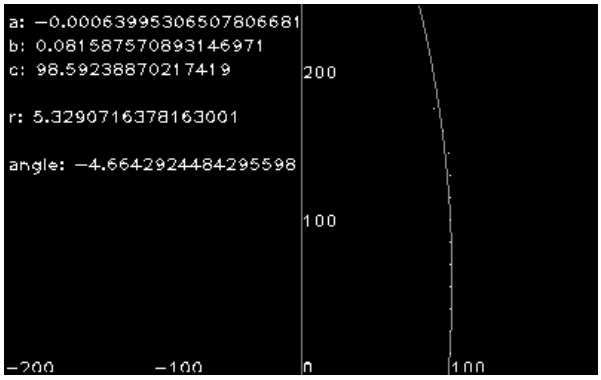
(a) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 1



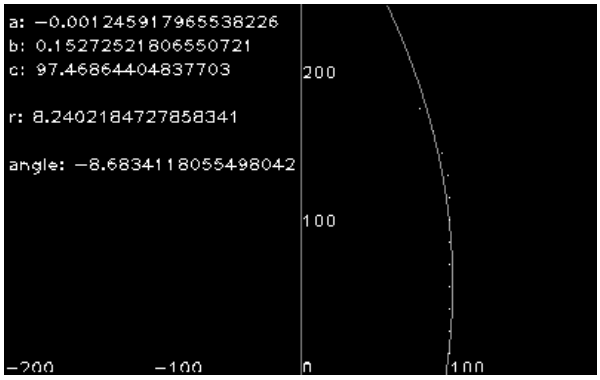
(b) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 2



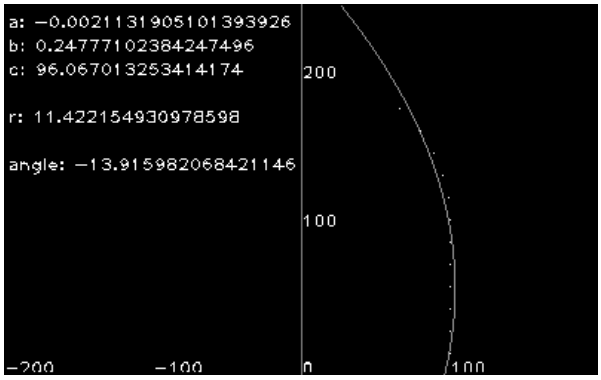
(c) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 3



(d) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 4

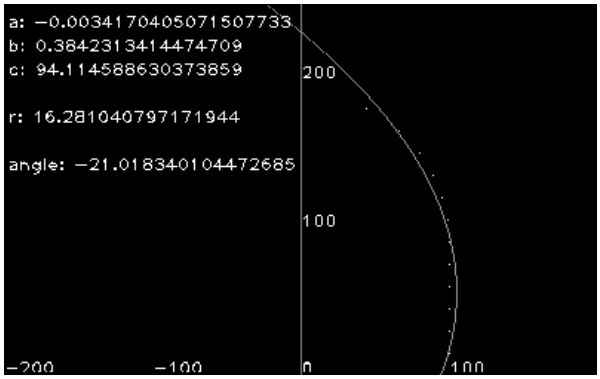


(e) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 5

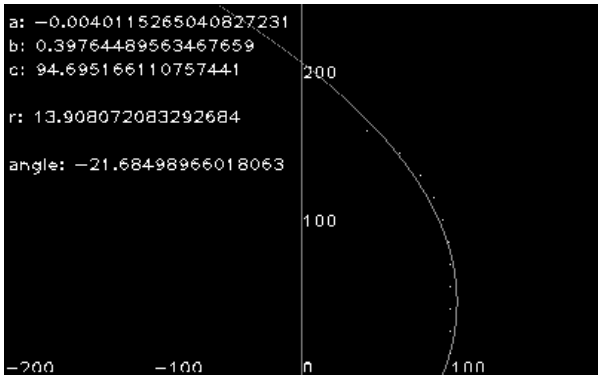


(f) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 6

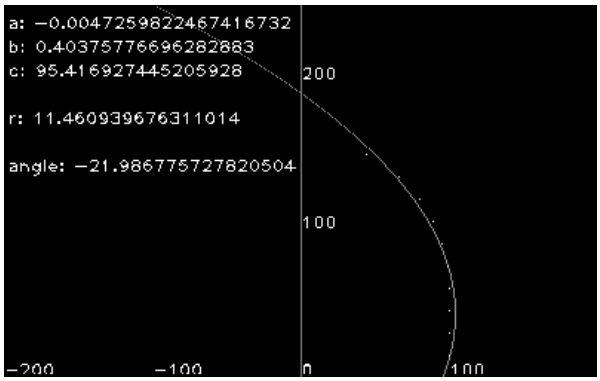
Abb. F.1: Approximation einer geraden Zufahrt auf eine enge Kurve mit einem quadratischem Polynom, Sequenz 1-6



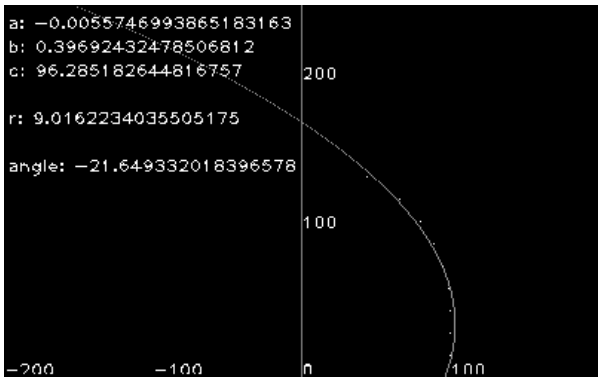
(a) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 7



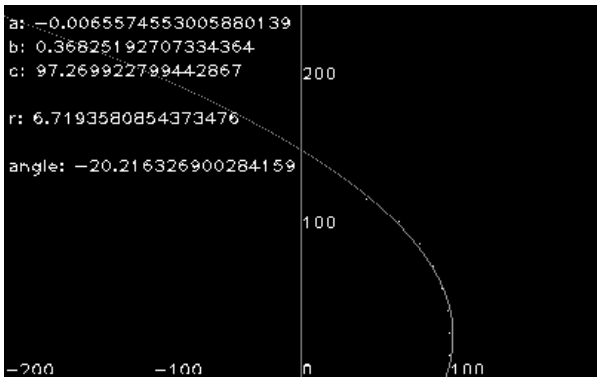
(b) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 8



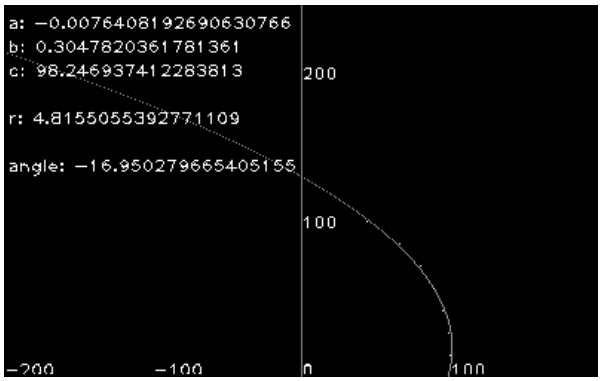
(c) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 9



(d) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 10



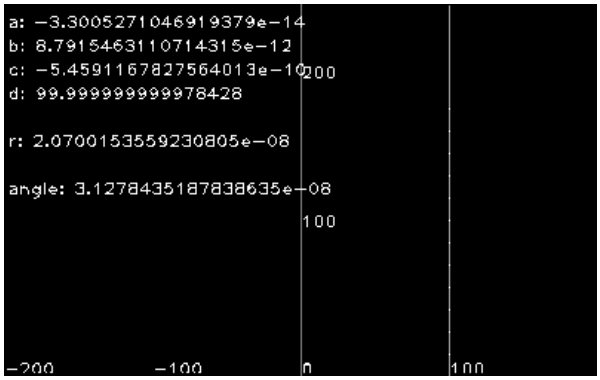
(e) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 11



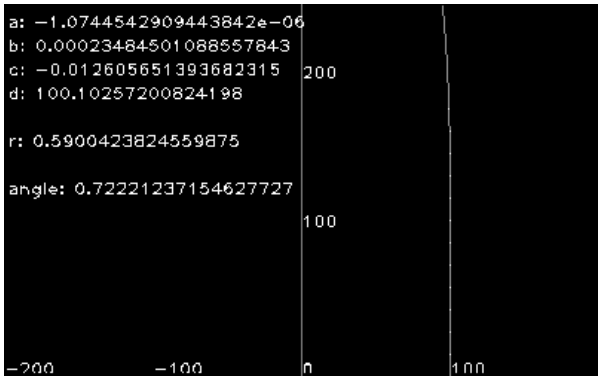
(f) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 12

Abb. F.2: Approximation einer geraden Zufahrt auf eine enge Kurve mit einem quadratischem Polynom, Sequenz 7-12

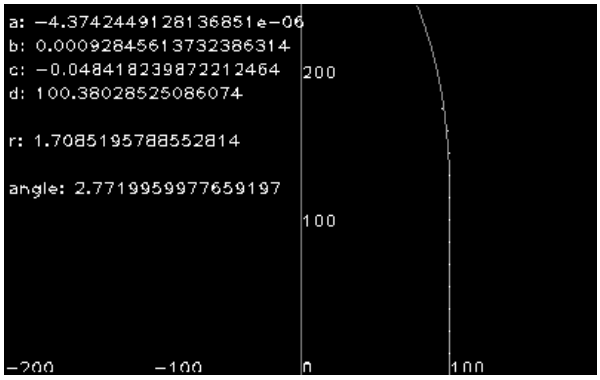
F.2 Bildsequenz des Polynom 3. Grades bei Zufahrt auf eine enge Kurve



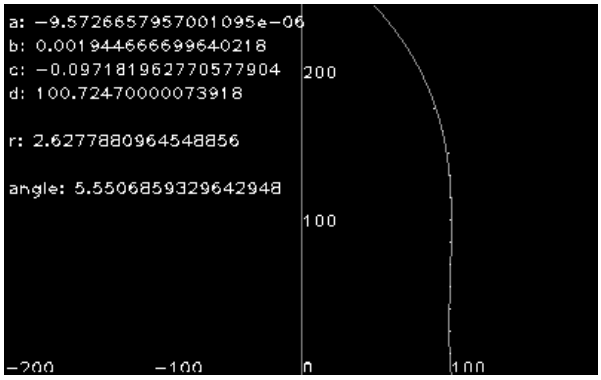
(a) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 1



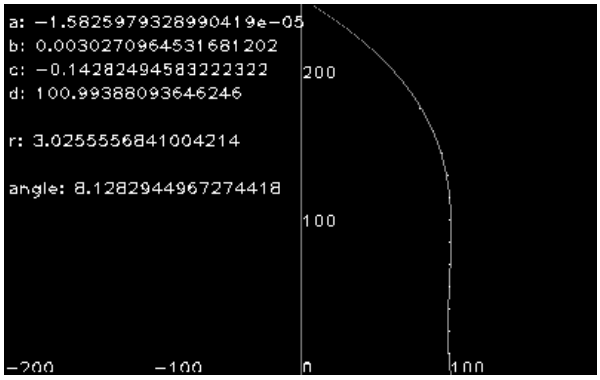
(b) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 2



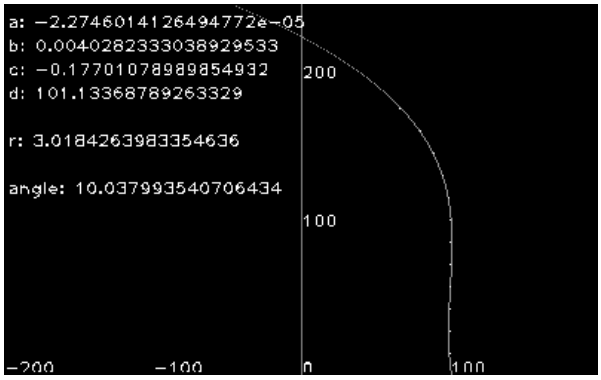
(c) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 3



(d) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 4

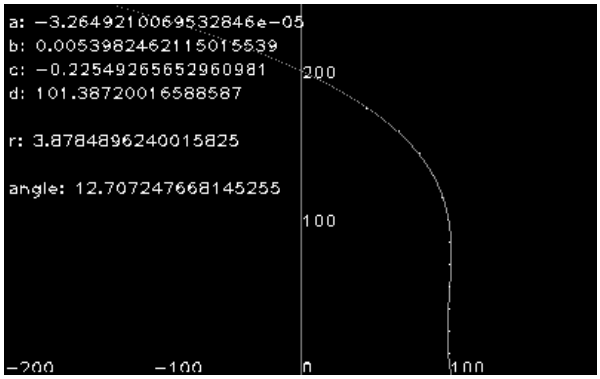


(e) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 5

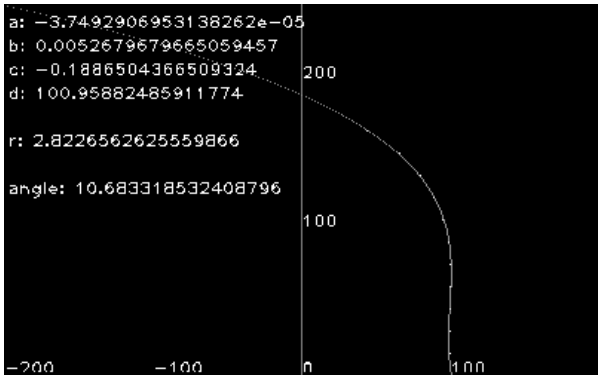


(f) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 6

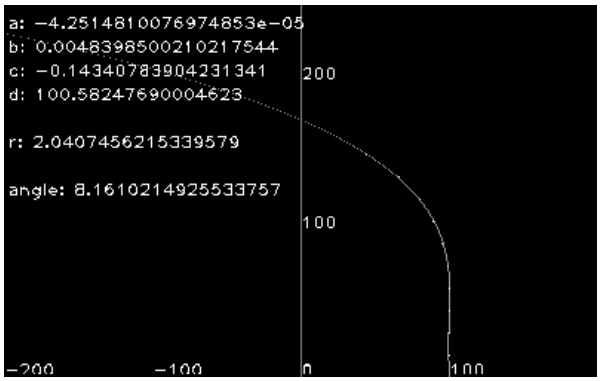
Abb. F.3: Approximation einer geraden Zufahrt auf eine enge Kurve mit einem kubischen Polynom, Sequenz 1-6



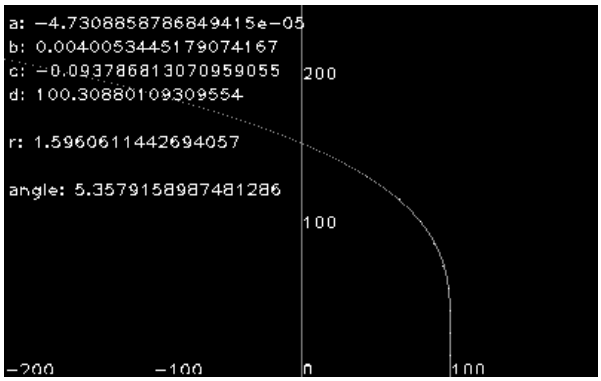
(a) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 7



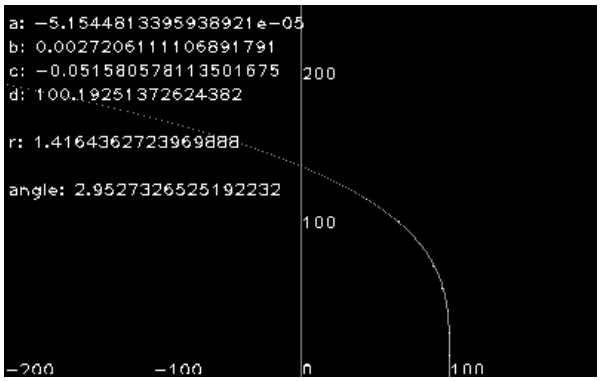
(b) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 8



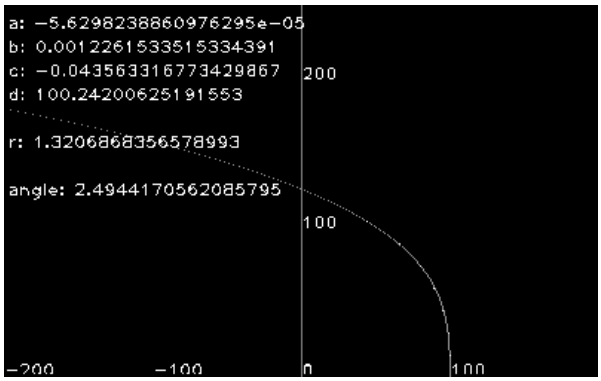
(c) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 9



(d) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 10



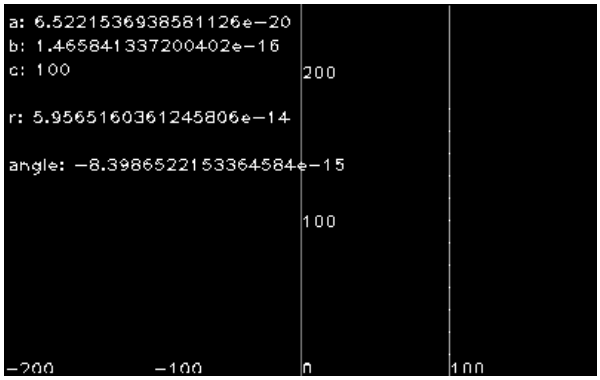
(e) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 11



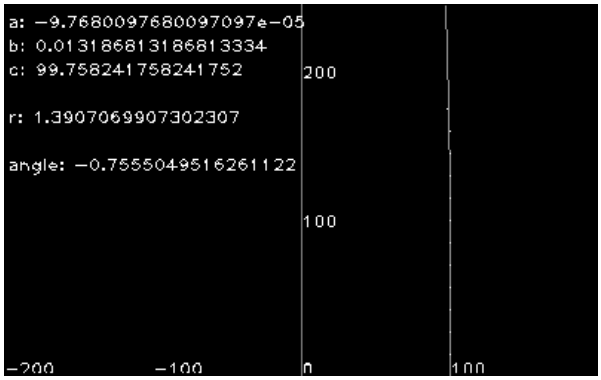
(f) Zufahrt auf Kurve, Innenradius 1 Meter, Bild 12

Abb. F.4: Approximation einer geraden Zufahrt auf eine enge Kurve mit einem quadratischem Polynom, Sequenz 7-12

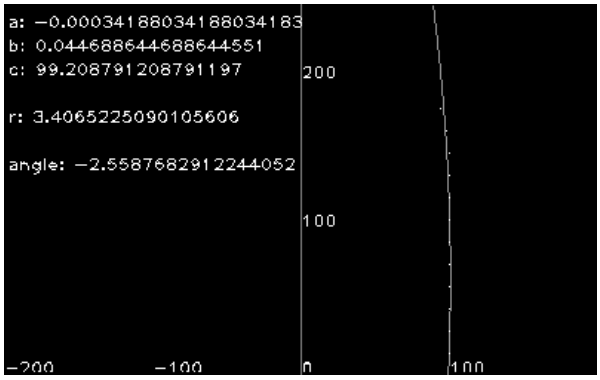
F.3 Bildsequenz des Polynom 2. Grades bei Zufahrt auf eine S-Kurve



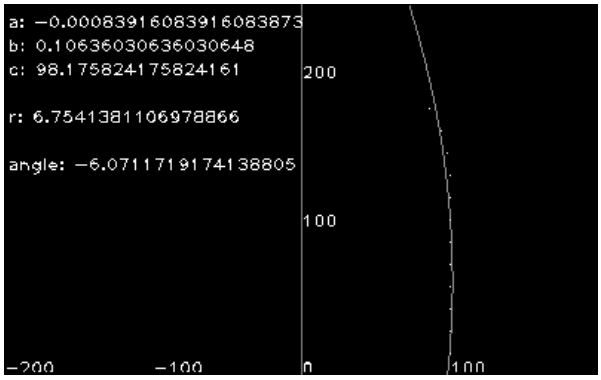
(a) Zufahrt auf S-Kurve, Bild 1



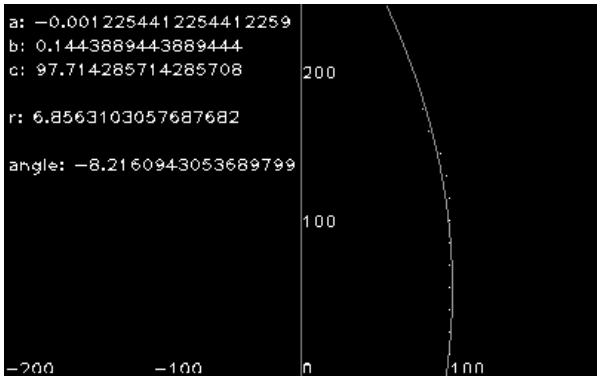
(b) Zufahrt auf S-Kurve, Bild 2



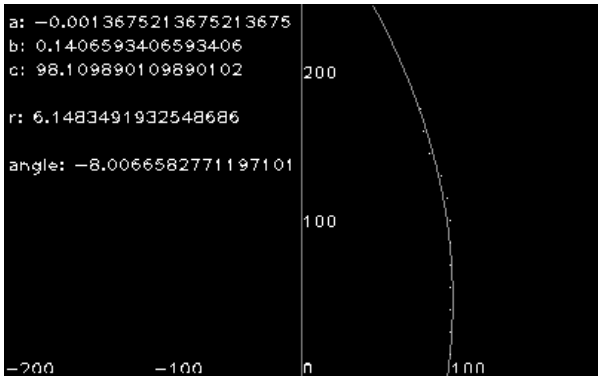
(c) Zufahrt auf S-Kurve, Bild 3



(d) Zufahrt auf S-Kurve, Bild 4

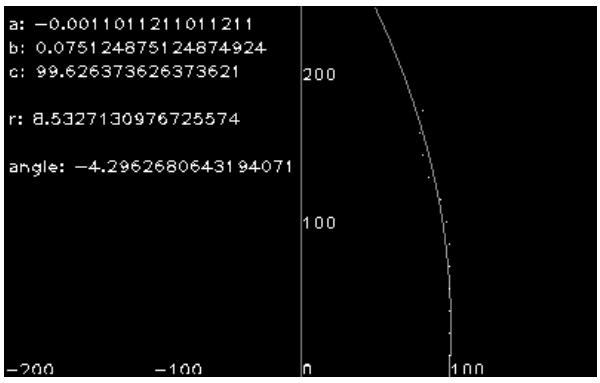


(e) Zufahrt auf S-Kurve, Bild 5

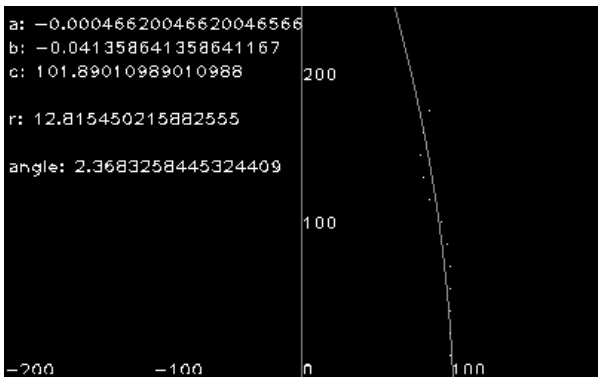


(f) Zufahrt auf S-Kurve, Bild 6

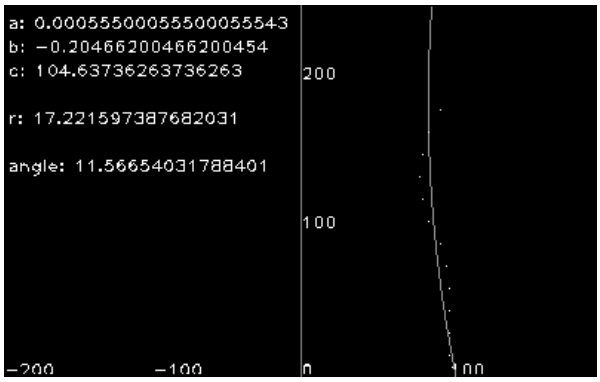
Abb. F.5: Approximation einer geraden Zufahrt auf eine S-Kurve mit einem quadratischem Polynom, Sequenz 1-6



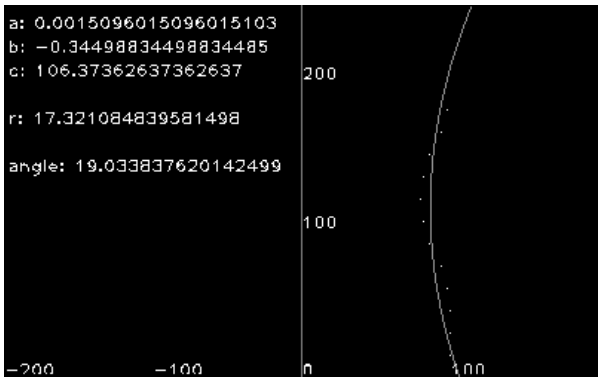
(a) Zufahrt auf S-Kurve, Bild 7



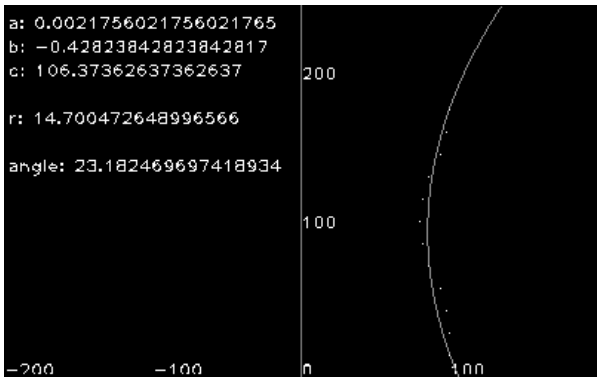
(b) Zufahrt auf S-Kurve, Bild 8



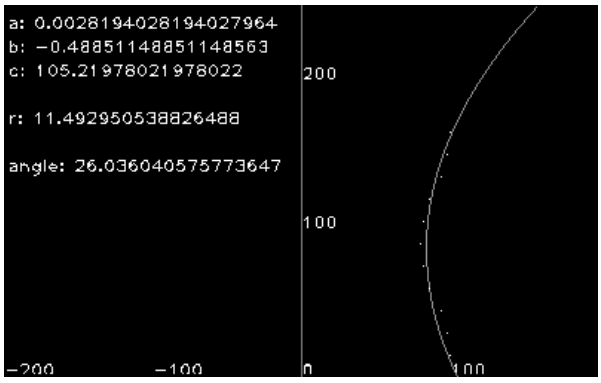
(c) Zufahrt auf S-Kurve, Bild 9



(d) Zufahrt auf S-Kurve, Bild 10



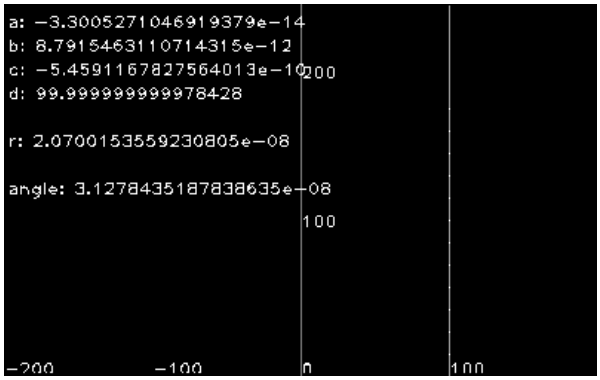
(e) Zufahrt auf S-Kurve, Bild 11



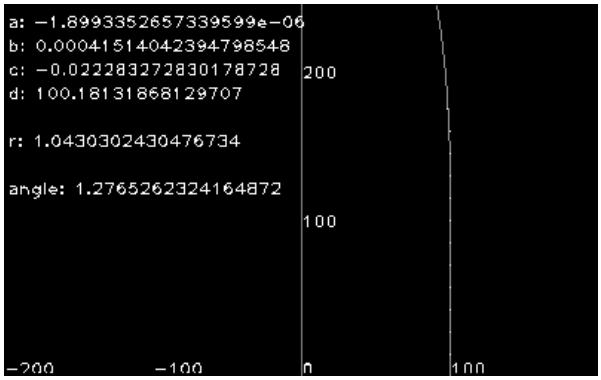
(f) Zufahrt auf S-Kurve, Bild 12

Abb. F.6: Approximation einer geraden Zufahrt auf eine S-Kurve mit einem quadratischem Polynom, Sequenz 7-12

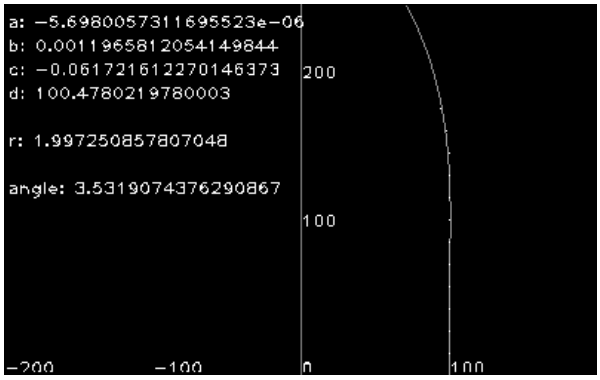
F.4 Bildsequenz des Polynom 3. Grades bei Zufahrt auf eine S-Kurve



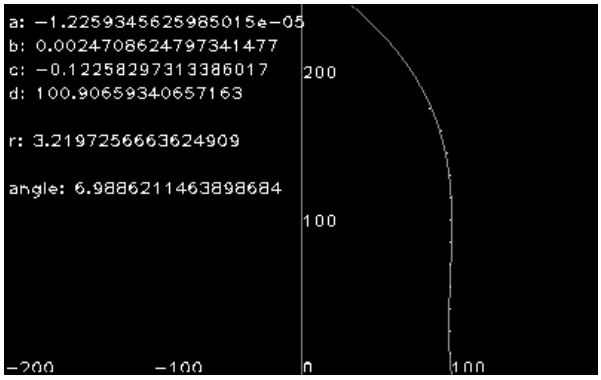
(a) Zufahrt auf S-Kurve, Bild 1



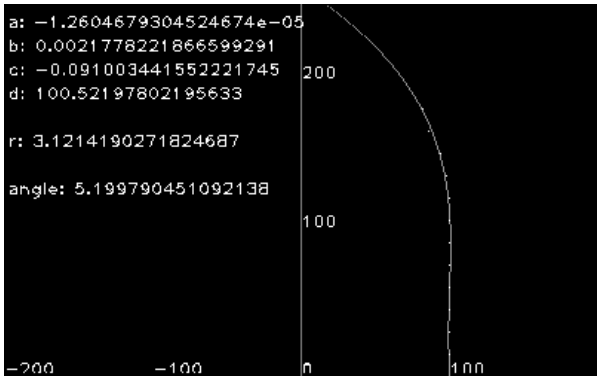
(b) Zufahrt auf S-Kurve, Bild 2



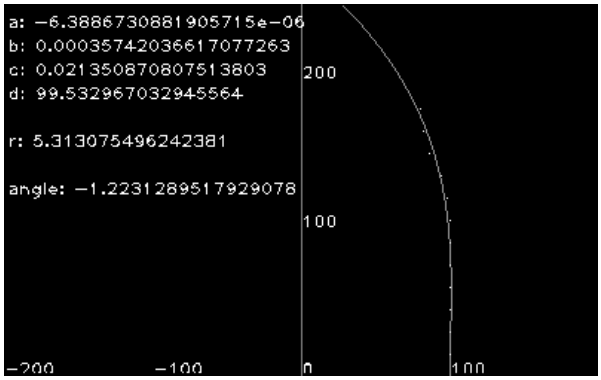
(c) Zufahrt auf S-Kurve, Bild 3



(d) Zufahrt auf S-Kurve, Bild 4

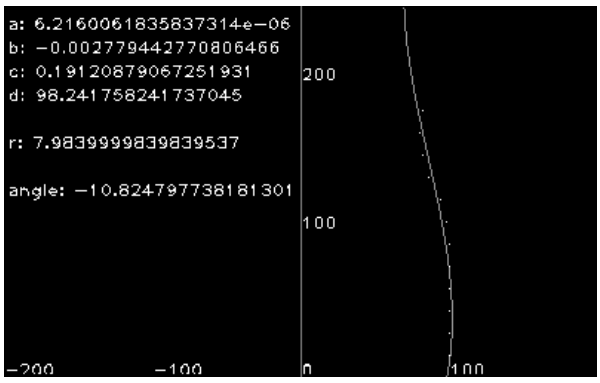


(e) Zufahrt auf S-Kurve, Bild 5

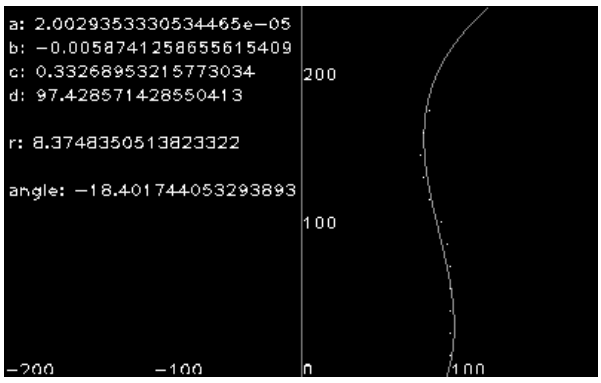


(f) Zufahrt auf S-Kurve, Bild 6

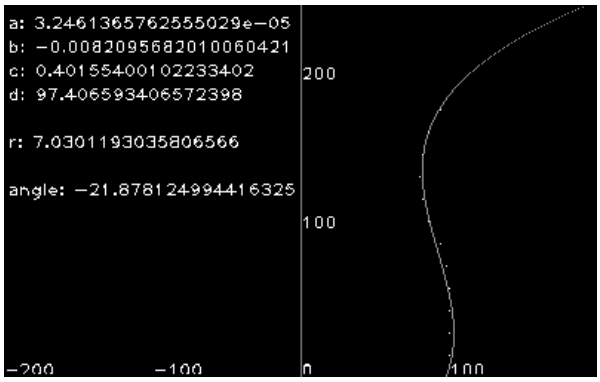
Abb. F.7: Approximation einer geraden Zufahrt auf eine S-Kurve mit einem kubischen Polynom, Sequenz 1-6



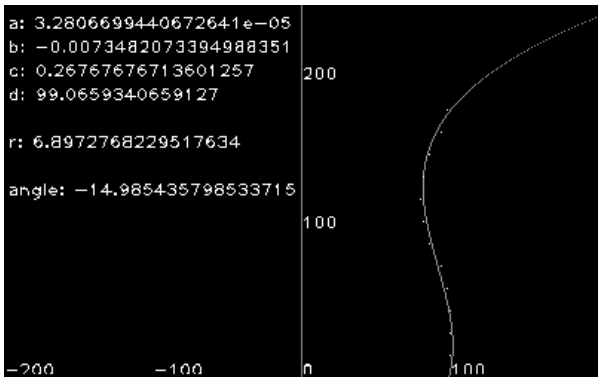
(a) Zufahrt auf S-Kurve, Bild 7



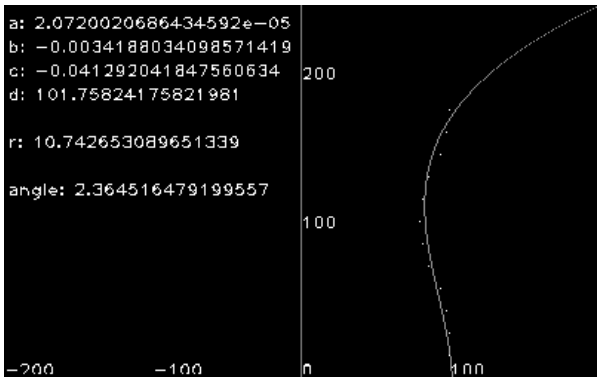
(b) Zufahrt auf S-Kurve, Bild 8



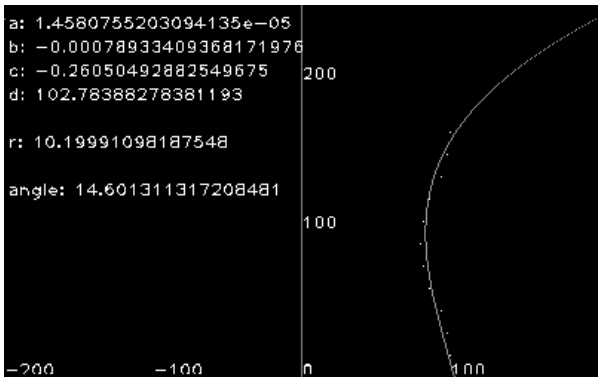
(c) Zufahrt auf S-Kurve, Bild 9



(d) Zufahrt auf S-Kurve, Bild 10



(e) Zufahrt auf S-Kurve, Bild 11



(f) Zufahrt auf S-Kurve, Bild 12

Abb. F.8: Approximation einer geraden Zufahrt auf eine S-Kurve mit einem kubischen Polynom, Sequenz 7-12

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §22(4) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 05. Dezember 2008	
Ort, Datum	Unterschrift