Abschlussbericht Projektseminar Echtzeitsysteme

Gruppe Crash Test Dummies
Projektseminararbeit eingereicht von
Kai Cui, Feiyu Chang, Regis Fayard, Lars Semmler, David Botschek
am 7. März 2019

TECHNISCHE UNIVERSITÄT DARMSTADT

Fachgebiet Echtzeitsysteme

Elektrotechnik und Informationstechnik (FB18)

Zweitmitglied Informatik (FB20)

Prof. Dr. rer. nat. A. Schürr
Merckstraße 25
64283 Darmstadt

www.es.tu-darmstadt.de

Gutachter: Stefan Tomaszek
Betreuer: Stefan Tomaszek

Erklärung zur Projektseminararbeit

Hiermit versichere ich, die vorliegende Projektseminararbeit selbstständig und ohne Hilfe Dritter angefertigt zu haben. Gedanken und Zitate, die ich aus fremden Quellen direkt oder indirekt übernommen habe, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen und wurde bisher nicht veröffentlicht.

Ich erkläre mich damit einverstanden, dass die Arbeit auch durch das Fachgebiet Echtzeitsysteme der Öffentlichkeit zugänglich gemacht werden kann.

Darmstadt, den 7. März 2019

_____

(Kai Cui, Feiyu Chang, Regis Fayard, Lars Semmler, David Botschek)

Inhaltsverzeichnis

1 Einführung **1**

2 Projektorganisation **2**

3 Protokoll **3**

4 Regelansätze **4**
4.1 PID-Regelungsansatz 4
4.2 Modellprädiktiver Regelungansatz 6

5 Schilderkennung **12**

6 Auswertung **13**

7 Konklusion **14**

1 Einführung

In der Automobilindustrie wird immer mehr auf autonomes Fahren gesetzt. Hier werden in der Entwicklung ständig neue Meilensteine erreicht, sodass ein Fahren ohne eine Person am Steuer immer realistischer wird.

Eine kleine Einführung will hier das Projektseminar Echtzeitsysteme der TU Darmstadt bieten. Anhand eines Modellautos werden Ansätze realitätsnaher Algorithmen diskutiert und ausprobiert. Die Kommunikation mit dem Auto geschieht mit Hilfe des Programms Robot Operating System (ROS). Mit dieser Methode testet auch der Kooperationspartner „Fachgebiet Fahrzeugtechnik (FZD)“ Lösungen am echten Auto, was die Relevanz dieses Projektseminars unterstreicht.

Ein weiterer, nicht weniger wichtiger Schwerpunkt des Seminars ist die erfolgreiche Planung und Durchführung der Gruppenarbeit. Die 5 Mitglieder der einzelnen Gruppen werden aus verschiedenen Vertiefungsrichtungen zugeteilt, damit jeder eigenes Knowhow mitbringen kann. Die verschiedenen Rollen und Verantwortlichkeiten werden im Team erprobt.

Fahrzeug

Dem Fahrzeug stehen verschiedene Sensoren wie der Ultraschall-, Gyro-, und Hallsensor sowie eine Kinect-Kamera zur Verfügung um das Umfeld möglichst realitätsnah wahrzunehmen. Durch die Kamera kann man sowohl auf ein Farbbild als auch auf ein Tiefenbild zugreifen was eine präzise Bildverarbeitung ermöglicht.

Aufgaben

Die verschiedenen Fortschritte wurden nach einzelnen Aufgaben gestaffelt. Das Abfahren eines Rundkurses anhand der Analyse zweier Linien stellte die Basis- und Pflichtaufgabe dar. Weitere Aufgaben durften selbst vorgeschlagen werden, als mögliche Beispiele wurden Spurwechsel, Hinderniserkennung und Verkehrsschilderkennung vorgeschlagen. Unsere Gruppe entschied sich für Spurwechsel und Erkennung der Schilder. Zum Projektumfang zählten aber genauso auch das Festlegen von und Prüfen in Testszenarien um die Funktionalität und Zuverlässigkeit autonomer Steärungen sicherzustellen.

2 Projektorganisation

Für einen reibungslosen Ablauf haben wir zu Beginn geplant, wie wir unsere Kommunikation gestalten wollen und konnten mit Hilfe unten aufgführter Tools eine erfolgsversprechende Roadmap erzielen.

Die Seminarorganisation schrieb ein regelmäßiges Treffen mit dem Seminarleiter vor um Hilfestellungen und Tipps zu ermöglichen. Vom Fachgebiet Fahrzeugtechnik wurde alle zwei Wochen eine Fragerunde zu regelungstechnischen Problemen angeboten. Dieses Treffen diente auch als Austausch mit den anderen Gruppen.

In unsere Kleingruppe entschieden wir uns dazu, einmal wöchentlich zusammen am Auto zu arbeiten um gegenseitiges Helfen zu ermöglichen und immer wieder den Stand abzuklären sowie den nächsten kurzen Abschnitt zu planen.

Den Rest der Woche wurde selbstständig an den neu zugeteilten Aufgaben weitergearbeitet. Bei akuten Fragen und Problemen fand ebenso eine ständige Kommunikation über WhatsApp statt.

Trello: Damit wir die Aufgaben präzise und strukturiert festhalten konnten haben wir uns für das Organisations-Tool Trello entschieden. Hiermit konnten wir durch Anlegen von Listen unsere Punkte in „ToDo“, „Meeting“, „Gemacht“ und „Aufgaben für die Zukunft“ aufteilen.

Github: Für die Synchronisierung des Programmcodes haben wir das Versionsverwaltungsprogramm GitHub genutzt. Dadurch konnten wir komfortabel den Programmiercode des Fahrzeugs austauschen sowie bei Fehlern auf alte Code-Zustände zurückgreifen.

Ebenso konnten wir beim Programmieren auf eine Reihe vorhandener libraries und Funktionen zurückgreifen.

OpenCV: Mit OpenCV [Rab] konnten wir die Bilder verarbeiten und die benötigten Informationen herausfiltern. Auch der Algorithmus zur sign recognition basiert auf einem Beispiel einer OpenCV-Demo [Lab]

quadprog++: Die library quadprog++ [Gas] stellt eine Funktion bereit, welche die Berechnung des MPC erleichtert.

Auch haben wir mit dem Programm doxygen [Hee] eine übersichtliche Auflistung all unserer Klassen mit Kommentaren erstellt.

3 Protokoll

Testfall	14.2.	21.2.	28.2.
Das Fahrzeug fährt rechts und links herum autonom ohne die Fahrbahnbegrenzung zu berühren.	✗	✓	✓
Das Fahrzeug fährt auf einer Fahrspur autonom ohne die Spur zu verlassen.	✓	✓	✓
Das Fahrzeug fährt eine Runde unter 30s.	✓ 25*s*	✓ 19*s*	✓ 19*s*
Das Fahrzeug kommt von der Fahrbahn ab und muss daraufhin anhalten.	✗	✗	✓
Das Fahrzeug startet mittig auf der Fahrbahn im 45 Winkel zur Fahrbahnbegrenzung	✓	✓	✓
Das Fahrzeug startet mittig auf der Fahrbahn im 60 Winkel zur Fahrbahnbegrenzung	✗	✓	✓
Bei einem Stoppschild hält das Fahrzeug für 3s an und fährt danach weiter.	✗	✓	✓
Bei einem Geschwindigkeitsschild fährt das Fahrzeug ab dem Schild mit verringerter Geschwindigkeit weiter.	✗	✓	✓
Bei einem Fahrbahnverengung Schild wechselt das Fahrzeug die Fahrspur.	✗	✓	✓
Schilderkennung und autonomes Fahren muss gleichzeitig ohne wahrnehmbare Beeinträchtigung funktionieren.	✗	✗	✓

4 Regelansätze

4.1 PID-Regelungsansatz

Unser erster Ansatz, das Fahrzeug autonom fahren zu lassen basierte auf einem PID-Regler. Dieses Verfahren lässt sich in 2 große Schritte aufteilen.

Bildverarbeitung: Für die Bildverarbeitung wird das Umfeld durch die Farbkamera der Kinect aufgenommen und in den HSV-Farbraum konvertiert. Zum Erkennen der Fahrbahnbegrenzung wird das Bild entsprechend nach grünen Punkten durchsucht und gefiltert. Um das entstandene Rauschen zu reduzieren wird zusätzlich noch ein Medianfilter verwendet. Das entstandene Bild wird in einem 2D-Array gespeichert, wobei die linke untere Ecke als Ursprung verwendet wird.

Für einen Spurwechsel ist das Bild der Kinect besonders in Kurven zu schmal, daher wird hier die mittlere Fahrbahnmarkierung genutzt. Hier wird das Bild nach rosafarbenen Punkten gefiltert. Da die mittlere Linie jedoch nicht durchgängig verläuft wird durch das Verfahren der Linearisierung digital eine durchgängige Markierung erzeugt. Zuerst wird der unterste weiße Punkt erfasst und in p_0 abgespeichert.

$$\mathbf{p}_0 \begin{pmatrix} x_0 & y_0 \end{pmatrix} \tag{1}$$

Jeder weitere weiße Punkt wird dann in p_i gesichert.

$$\mathbf{p}_i \begin{pmatrix} x_i & y_i \end{pmatrix} \tag{2}$$

Durch die erkannten Punkte kann die Steigung k_i errechnet werden.

$$\mathbf{k_i} = \frac{\mathbf{y_i} - \mathbf{y_0}}{\mathbf{x_i} - \mathbf{x_0}} \tag{3}$$

Zuletzt wird daraus dann der Durschnittswert aller Steigungen S berechnet.

$$\mathbf{S} = \frac{\mathbf{1}}{\mathbf{m}} \sum \mathbf{k_i} \tag{4}$$

Mit dieser Steigung werden die Lücken zwischen den erkannten Linienabschnitten ergänzt.

Regelkreis: Für die Steuerung des Fahrzeugs wird die Differenz zwischen der aktuellen Stellung des Fahrzeugs und der geplanten Richtung verwendet. Relativ zu dieser Abweichung wird das Steering-Level entsprechend gesetzt.

Um die Analyse der Geschwindigkeit zu vereinfachen, wird hierzu lediglich eine Dimension betrachtet, da die anderen Geschwindigkeitsvektoren dazu relativ klein und vernachlässigbar sind. Die Orientierung der Geschwindigkeit wird als B(t) ausgegeben.

$$\mathbf{B(t)} = \begin{pmatrix} x_b(t) & y_b(t) \end{pmatrix} \tag{5}$$

Als Sollwert A(t) wird die aufgenommene und bearbeitete Strecke verwendet. Die Abweichung zwischen Soll- und Ist-Wert wird aufintegriert und als Eingabewert e(t) für die PID-Regelung verwendet, wodurch das passende Steering-Level u(t) gesetzt werden kann.

$$\mathbf{A(t)} = \begin{pmatrix} x_a(t) & y_a(t) \end{pmatrix} \tag{6}$$

$$\mathbf{e(t)} = \sum_{\mathbf{y=0}}^{\mathbf{n}} | \mathbf{x_A} - \mathbf{x_B} | \tag{7}$$

$$\mathbf{u(t)} = \mathbf{K_p} * \mathbf{e(t)} + \mathbf{K_D} * \dot{\mathbf{e}}\mathbf{(t)} \tag{8}$$

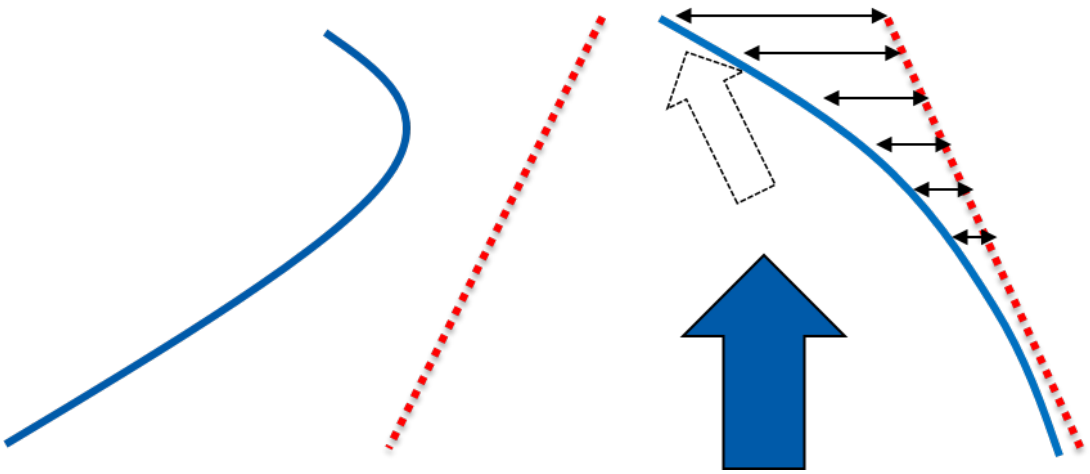

Abbildung 4.1: PID-Regeldifferenz, Solllinie(rot), Lanepixel(blau), Abweichungen(schwarz)

4.2 Modellprädiktiver Regelungansatz

Ein zweiter experimenteller Ansatz, der im Rahmen des Projektseminars verfolgt wurde, ist die sogenannte lineare modellprädiktive Regelung. Bei dieser wird anhand eines linearen, diskretisierten Fahrzeugmodells über ein quadratisches Optimierungsproblem eine Lösung für die optimalen Lenkwinkel berechnet. Diese sind optimal in dem Sinne, dass eine Linearkombination der mittleren quadratischen Abweichungen des prädiktierten Systemzustands und Stellgröße zu einer vorgegebenen Solltrajektorie über einen endlichen Zeithorizont hinaus minimiert wird.

Ein großer Vorteil dieses Verfahrens ist, dass es automatisch Stellgrößenbeschränkungen beachtet und es somit nicht zu Prädiktionsfehlern durch unbeschränkte Stellgrößen kommt. Durch die Methode erhält man zugleich den gesamten Stellgrößenverlauf für den eingestellten Zeithorizont, sodass es außerdem möglich ist, über längere Zeit ohne Neuberechnung an einer Solltrajektorie entlang zu fahren, vorausgesetzt das linearisierte Modell ist für die gegebene Zustandstrajektorie ausreichend genau.

Zunächst bringen wir das Problem in die Form, die für die lineare modellprädiktive Regelung benötigt wird. Dabei orientieren wir uns an [Hov04] sowie an Vorarbeiten von Herrn Eric Lenz am FG RTM. Wir gehen aus vom Querführungsmodell in Gleichung 9 von Herrn Eric Lenz am FG RTM, welches bereits das Ackermannmodell auf eine Raumdimension reduziert:

$$\begin{pmatrix} \dot{y} \\ \dot{\varphi}_K \end{pmatrix} = \begin{pmatrix} 0 & v \\ 0 & 0 \end{pmatrix} \begin{pmatrix} y \\ \varphi_K \end{pmatrix} + \begin{pmatrix} v \cdot \frac{l_H}{l} \\ \frac{v}{l} \end{pmatrix} \varphi_L^* = \mathbf{Ax} + \mathbf{Bu} \tag{9}$$

Das Modell wird per Hand mit der Abtastperiode T diskretisiert zu

$$\mathbf{A_d} = e^{\mathbf{A}T} = \begin{pmatrix} 1 & vT \\ 0 & 1 \end{pmatrix} \tag{10}$$

$$\mathbf{B_d} = \int_0^T e^{\mathbf{A}\tau} \mathbf{B} d\tau = \begin{pmatrix} \frac{vTl_h}{l} + \frac{v^2T^2}{2l} \\ \frac{vT}{l} \end{pmatrix} \tag{11}$$

$$\begin{pmatrix} y(k+1) \\ \varphi_K(k+1) \end{pmatrix} = \begin{pmatrix} 1 & vT \\ 0 & 1 \end{pmatrix} \begin{pmatrix} y(k) \\ \varphi_K(k) \end{pmatrix} + \begin{pmatrix} \frac{vTl_h}{l} + \frac{v^2T^2}{2l} \\ \frac{vT}{l} \end{pmatrix} \varphi_L^*(k) \tag{12}$$

Es sei nun eine Trajektorie nach den Gleichungen 13, 14 gegeben, deren Erzeugung wir an späterer Stelle erläutern.

$$x_{ref} = \begin{pmatrix} x_{ref,1} & x_{ref,2} & \dots & x_{ref,n-1} & x_{ref,n} \end{pmatrix}^T \tag{13}$$

$$u_{ref} = \begin{pmatrix} u_{ref,0} & u_{ref,1} & \dots & u_{ref,n-2} & u_{ref,n-1} \end{pmatrix}^T \tag{14}$$

Dann definieren wir

$$\hat{Q} = 100 \cdot diag(1, 0.001, \cdots, 1, 0.001); \quad \hat{P} = \mathbf{I}_{n,n}; \quad -U_{min} = U_{max} = 21^o$$

$$x_0 = \begin{pmatrix} y(k) \\ 0 \end{pmatrix}; \quad A = \begin{pmatrix} 1 & vT \\ 0 & 1 \end{pmatrix}; \quad B = \begin{pmatrix} \frac{vTl_h}{l} + \frac{v^2T^2}{2l} \\ \frac{vT}{l} \end{pmatrix}$$

$$\chi_0 = \begin{pmatrix} A \\ A^2 \\ \vdots \\ A^{n-1} \\ A^n \end{pmatrix} x_0 + \begin{pmatrix} B & 0 & \dots & 0 & 0 \\ AB & B & \ddots & \vdots & \vdots \\ \vdots & \vdots & \ddots & 0 & 0 \\ A^{n-2}B & A^{n-3}B & \dots & B & 0 \\ A^{n-1}B & A^{n-2}B & \dots & AB & B \end{pmatrix} u_{ref} - x_{ref} \tag{15}$$

Wir möchten nun die quadratischen Zustandsabweichungen und Stellgrößen über den endlichen Zeithorizont n unter Beachtung der Stellgrößenbeschränkung und Systemdynamik optimieren.

$$\begin{aligned} \min_v f(x,u) = \sum_{t=0}^{n-1} \{ & (x_i - x_{ref,i})^T Q (x_i - x_{ref,i}) \\ & + (u_i - u_{ref,i})^T P (u_i - u_{ref,i}) \} \\ & + (x_n - x_{ref,n})^T Q (x_n - x_{ref,n}) \end{aligned}$$

unter den Nebenbedingungen

$$\begin{aligned} x_0 &= gegeben \\ x_i &= Ax_{i-1} + Bu_{i-1} \quad \text{for} \quad 1 \le i \le n \\ U_L &\le u_i \le U_U \quad \text{for} \quad 0 \le i \le n-1 \end{aligned}$$

Wir können dann das folgende äquivalente quadratische Optimierungsproblem für n Zeitschritte definieren [Hov04], welches die quadratischen Positionsabweichungen über n Zeitschritte vorwärts minimiert.

$$\min_v f(v) = 0.5 v^T \tilde{H} v + c^T v \tag{16}$$

$$= (x_0 - x_{ref,0})^T Q (x_0 - x_{ref,0}) + \chi_0^T \hat{Q} \chi_0 + 2\chi_0^T \hat{Q} \chi_v + \chi_v^T \hat{Q} \chi_v + v^T \hat{P} v \tag{17}$$

mit den Stellgrößen-Nebenbedingungen

$$\mathbf{I}_{n,n} v \ge \begin{pmatrix} U_{min} \\ \vdots \\ U_{min} \end{pmatrix} - u_{ref} \tag{18}$$

$$-\mathbf{I}_{n,n} v \ge -\begin{pmatrix} U_{max} \\ \vdots \\ U_{max} \end{pmatrix} + u_{ref} \tag{19}$$

Da wir keine Zustandsgrößenbeschränkung vorsehen, benötigen wir die restlichen Nebenbedingungen aus [Hov04] nicht. Dieses Optimierungsproblem wird für die gegebene Trajektorie on-line auf dem Fahrzeug gelöst, und der folgende optimale Stellgrößenverlauf wird abgespielt:

$$\varphi_{L,i} = \arctan(\varphi^*_{L,i}) = \arctan(u_i) = \arctan(v_i + u_{ref,i}) \tag{20}$$

Die Konvergenz, Optimalität und numerische Stabilität der Lösung des Optimierungsproblems der linearen modellprädiktiven Regelung ist hierbei durch die Verwendung von Methoden aus der quadratischen Programmierung, z.B. hier [GI83], sichergestellt.

Wir haben hier die Gewichtungsmatrizen $\hat{Q}, \hat{P}$ so gesetzt, dass näherungsweise lediglich die Positionsabweichung zur Solltrajektorie minimiert werden soll. Dies erlaubt es uns daher, als Solltrajektorie den Sollstellgrößenverlauf zu ignorieren und eine beliebige Sollwinkeltrajektorie vorzugeben, da Inkonsistenzen mit dem Sollzustandsverlauf kaum ins Gewicht fallen:

$$u_{ref} = \begin{pmatrix} 0 & 0 & \dots & 0 & 0 \end{pmatrix}^T \tag{21}$$

Als nächstes wird die Beschaffung der Sollzustandstrajektorie erläutert. Dazu bedienen wir uns der Kamera, dessen Farbdaten zunächst mithilfe einer inversen Perspektiventransformation (IPM) auf ein Birdeye-View gebracht werden. Beispielhaft ist ein bearbeitetes, transformiertes Bild in Abbildung 4.3 zu sehen.

Das Bild wird in horizontale Streifen einer vordefinierten Breite unterteilt. Anschließend werden die Streifen nach den Fahrbahnbegrenzungsfarben gefiltert und Spaltenhistogramme gebildet. Die Histogrammmaxima am nächsten zur Mitte des Bildes mit einem vordefinierten Mindestabstand werden selektiert. In diesem Fall also bis zu zwei Seitenstreifenpunkte und bis zu ein Mittellinienpunkt. Diese sind in Abbildung 4.2 als schwarze Kreise eingezeichnet.

Anschließend werden diese Punkte verwendet, um sinnvolle Wegpunkte zu definieren. Diese sind in der Abbildung 4.2 als grüne Kreise eingezeichnet. Wegpunkte werden priorisiert zwischen einen Mittellinienpunkt und den Seitenstreifenpunkt der entsprechenden Fahrbahnseite gelegt. Falls das nicht möglich ist, z.B. weil einer der Punkte nicht existiert, legen wir ihn $\frac{1}{4}$ oder $\frac{3}{4}$ zwischen die Seitenstreifenpunkte je nach Fahrbahnseite. Falls auch das nicht möglich ist, können zusätzliche Punkte aus den bereits existierenden inferiert werden, falls mindestens ein Seitenstreifenpunkt existiert.

Mithilfe dieser Wegpunkte kann man nun eine Trajektorie x_{ref} erzeugen.In Abbildung 4.2 ist die blaue Linie eine quadratische Trajektorie, die an die grünen Wegpunkte gefittet wird. Die Trajektorie scheint zunächst gut an die Fahrbahn angepasst zu sein. Man erkennt, dass die Solltrajektorie nicht einhaltbar ist, da die Querabweichung sich aus dem Unterschied zwischen polynomiell extrapolierter Solltrajektorie und Roboterposition zusammensetzt und nicht bei null beginnt. Dies ist kein Problem, da lediglich der quadratische Fehler über die nächsten Zeitschritte minimiert wird.

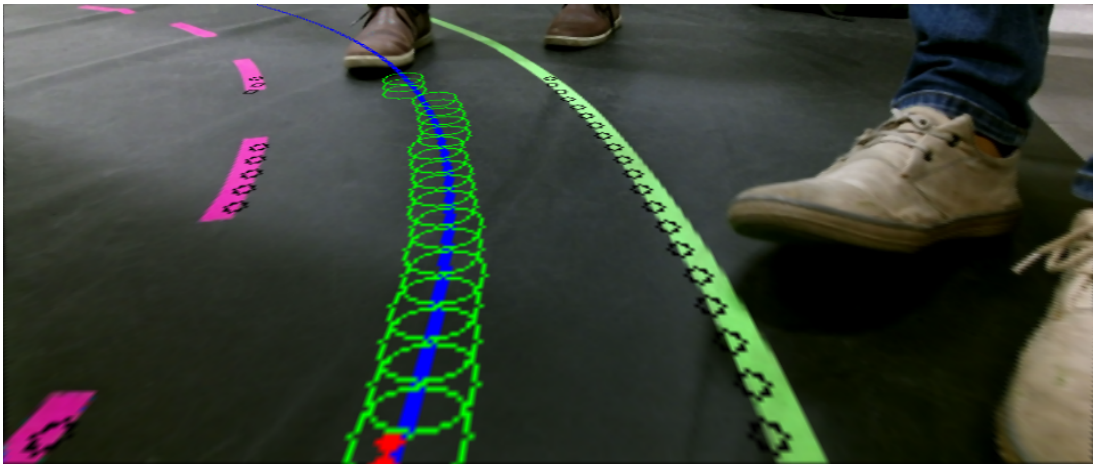

Abbildung 4.2: Bearbeitetes, rücktransformiertes Bild mit quadratischer Trajektorie. Sichtbar sind Wegpunkte (grüne Kreise), die Solltrajektorie (blaue Linie), die prädiktierte Trajektorie (rote Punkte) und Lanepunkte (schwarze Kreise)

Ein Problem lässt sich in Abbildung 4.3 jedoch feststellen: Die Trajektorien sind aufgrund der polynomiellen Extrapolation nicht stationär und sehr variant. Zusätzlich kommt hinzu, dass die Regressionsmethode nicht rotationsinvariant ist, denn eine Rotation der Wegpunkte führt aufgrund der vertikalen Polynomausrichtung zu einer völlig anderen Trajektorie. Diese Umstände führen dazu, dass die Regelung versagt.

Eine einfache Lösung ist es, nicht zu interpolieren, sondern wie in Abbildung 4.4 zu sehen eine direkte lineare Trajektorie zum nächstliegenden Wegpunkt zu setzen. Dadurch wird die Trajektorie deutlich stationärer und robuster, sodass die Regelung wie erwartet funktioniert. Dasselbe Problem hat man jedoch im begrenzten Maße auch für die direkte lineare Trajektorie. Selbst wenn man auf Extrapolation verzichtet und ausreichend Mittelung verwendet, ist der unterste Bildpunkt aufgrund der auftretenden Rotation des gegenüber der Roboterposition orthogonal ausgerichteten Bildes auch für gerade Fahrstrecken variant, sobald eine Abweichung auftritt. Zusätzlich kommt es durch die Diskretisierung bei großen Zeitschritten zu Überschwingen im kontinuierlichen physikalischen System trotz Einhaltung der Zustandssollwerte im diskreten Modell.

Mit dieser Bildtrajektorie und den Bildtransformationsparametern lassen sich dann diskrete Trajektoriensollwerte $x_{ref,i}$ erzeugen, indem das kinematische Querführungsmodell angenommen wird: Für jeden Trajektorienzeitschritt wird eine konstante Vorwärtsbewegung abhängig von der Geschwindigkeit ausgeführt, um den nächsten Positionssollwert zu ermitteln.

Das größte Problem ist das nicht ausreichend genaue Modell. der MPC-Ansatz erlaubt noch weitere Verbesserungen, um besseres Regelverhalten zu erreichen und insbesondere hohe Geschwindigkeiten ausreichend genau zu regeln. Darauf wird in diesem Projektseminar aber aufgrund der abweichenden Zielsetzung verzichtet.

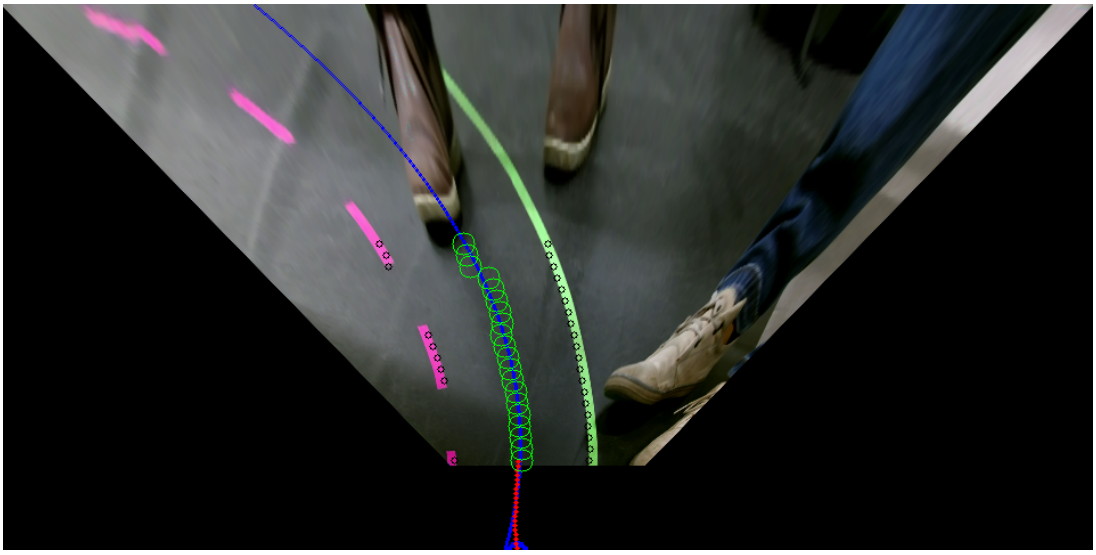

Abbildung 4.3: Transformierte und um Roboterposition erweiterte Version von Abbildung 4.2.

Es bietet sich zukünftig an, das Fahrzeugmodell durch ein nichtlineares Modell zu ersetzen und allgemeinere Optimierer zu verwenden. Damit landet man bei der nichtlinearen modelprädiktiven Regelung (NMPC), z.B. [AZ12], bei der ein anderer Modellansatz verfolgt werden muss, da das Querabweichungsmodell eine Linearisierung ist.

Weiter kann man das Modell durch ein genaueres Modell ersetzen. Erstens bietet es sich an, ein genaueres Modell als das Ackermann-Modell zu verwenden, um auch bei hohen Fahrgeschwindigkeiten ein ausreichend genaues Dynamikmodell zu erhalten. Zweitens kann man auch Modelle für die Servolenkung berücksichtigen, da diese nicht unendlich schnell einen Lenkwinkel einstellen kann.

Schließlich könnte man auch die Trajektoriengenerierung überarbeiten. Eine naheliegende Wahl, um auch mit Bildraten von unter 1 Hz auszukommen, ist die Verwendung der direkten Trajektorie bis zum ersten Wegpunkt und darauffolgende Verwendung der quadratischen Trajektorie. Da die quadratische Trajektorie in diesem Fall nicht mehr extrapoliert, bietet diese eine hervorragende Trajektorienform für eine Regelung mit wenigen Bildern.

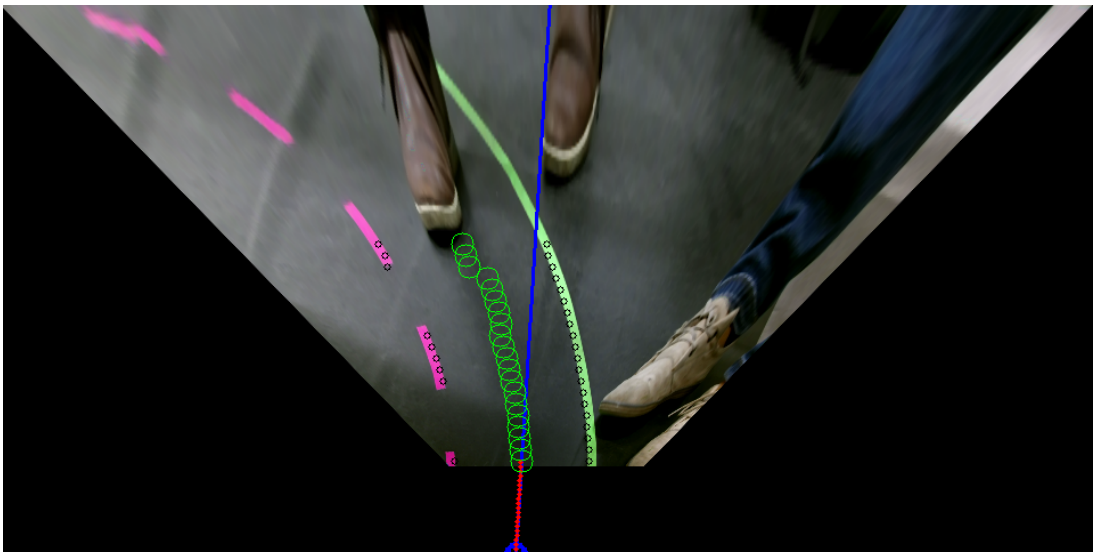

Abbildung 4.4: Bearbeitetes, untransformiertes Bild mit quadratischer Trajektorie. Sichtbar sind Wegpunkte (grüne Kreise), die Solltrajektorie (blaue Linie), die prädiktierte Trajektorie (rote Punkte), Lanepunkte (schwarze Kreise) und die Roboterposition relativ zum Bild (Anfangspunkt der prädiktierten Trajektorie, großer blauer Kreis)

5 Schilderkennung

6 Auswertung

7 Konklusion

Das Projektseminar stellte sich als eine sehr vielfältige und facettenreiche Veranstaltung heraus, bei der das eigentliche Ziel, eine Fahrzeugsteuerung zu programmieren, nur einen Teil der Erkenntnisse darstellt. Durch den längeren Zeitraum von einem Semester konnten wir uns intensiv mit Ansätzen des Autonomen Fahrens beschäftigen und konnten so mit direktem Testen des Codes auf eine spannende und praxisnahe Weise Vieles lernen. Auch war es sehr Wissen erweiternd, den Umgang mit der Roboter-Middlerware ROS zu erlernen. Dass dieses System auch in der Industrie eingesetzt wird, zeigt wie wichtig und aktuell diese Kompetenzen in der Arbeitswelt sind. Ebenso ermöglichte das Seminar einen guten Einblick in die verschiedenen Herausforderungen der Projektorganisation. Dadurch lernten wir, die einzelnen Schritte möglichst effizient aufzuteilen und wurden erfahrener in der Verwendung verschiedener Tools, die uns sehr unterstützten.

Wir konnten auf eine sehr umfangreiche und freundliche Projektbetreuung zurückgreifen, sodass wir für die verschiedenen Probleme immer einen Ansprechpartner hatten. Sehr hervorzuheben ist auch die freie Gestaltung der Aufgaben. Der genaue Zeitplan war uns überlassen, die Zusatzaufgabe durften wir uns selbst überlegen.

Dennoch kamen wir auch mit unseren vergleichsweise primitiven Berechnungen an die Leistungsgrenzen der Recheneinheit. Hier wäre es schön, wenn eine schnellere Version verbaut würde bzw. eine dezidierte Grafikeinheit zum Einsatz kommen würde, damit Bilder besser verarbeitet werden könnten, da diese Disziplin im Fokus stand. Auch kam es immer wieder zum Absturz des Betriebssystems des Fahrzeugs, was zu unerwünschten Unterbrechungen unserer Arbeit führte.

Literatur

[AZ12] Allgöwer, Frank ; Zheng, Alex: *Nonlinear model predictive control*. Bd. 26. Birkhäuser, 2012

[Gas] Gaspero, Luca D.: *quadprog++*. https://github.com/liuq/QuadProgpp/blob/master/src/QuadProg%2B%2B.hh

[GI83] Goldfarb, Donald ; Idnani, Ashok: A numerically stable dual method for solving strictly convex quadratic programs. In: *Mathematical programming* 27 (1983), Nr. 1, S. 1–33

[Hee] Heesch, Dimitri van: *doxygen*. http://www.doxygen.nl/

[Hov04] Hovd, Morten: A brief introduction to Model Predictive Control. In: *URL= http://www. itk. ntnu. no/fag/TTK4135/viktig/MPCkompendium% 20HOvd. pdf* (2004)

[Lab] Labbe, Mathieu: *findObject*. https://github.com/introlab/find-object/blob/master/src/FindObject.cpp

[Rab] Rabaud, Vincent: *OpenCV*. http://wiki.ros.org/vision_opencv