Projektseminar Echtzeitsysteme

AUDO - Autonomous Unmanned Driving Unit

Nikolas Ziegelmayer, Nils Wittig, Fabian Burger, Ramona Volz und Maike Latsch
Institut für Datentechnik | Fachgebiet Echtzeitsysteme

TECHNISCHE UNIVERSITÄT DARMSTADT

Inhaltsverzeichnis

Todo list

1 Einleitung

Im Gegensatz zu den vergangenen Jahren war es die Aufgabe dieses Projektseminars Echtzeitsysteme eine Steuerung zu entwickeln, die ein Modellauto innerhalb gegebener Spuren fahren lässt. Durch eine Kamera und verschiedene Sensoren wird die Umgebung erkannt und mittels Filterungen und Regelungen in Steuerbefehle für das Fahrzeug umgewandelt.

Weiter schreiben

2 Grundlagen

2.1 Modellauto

evtl. Hardwaregrundlagen

2.2 OpenCV

2.3 ROS - Robot Operating System

ROS ist ein Metabetriebssystem für Roboter, welches auf Linux basiert und in vielen Unternehmen für Steuerung von Robotern genutzt wird. Es stellt mehrere Pakete zur Verfügung, die einige nützliche Funktionen ermöglichen. Dazu gehört die Verteilung auf mehrere Systeme im Netzwerk, Paketverarbeitung und die Kommunikation zwischen den Nodes [1]. Die Kommunikation findet durch ein Publish-Subscribe-System statt. Die einzelnen Nodes publishen zu bestimmten Themen Nachrichten, deren Inhalt sich auf das Thema bezieht. So erhält man zum Thema `/uc_bridge/usr` über eine Subscription Nachrichten über die Werte des rechten Abstandssensors. Auch die Steuerung des Motors und des Lenkwinkels geschieht über das puplishen von Nachrichten.

2.4 PSES Packages

3 Organisation

3.1 Aufgabenverwaltung

3.2 Versionsverwaltung

4 Implementierung

4.1 Bildverarbeitung und Erkennung von Fahrspuren

Fabian

4.2 Erkennung von Kurven und Geraden

Fabian

4.3 Erkennung von Hindernissen

Fabian

4.4 Kollisionsvermeidung

Um in allen Fahrsituationen eine Beschädigung des Fahrzeugs zu vermeiden, wird eine Kollisionsvermeidung eingesetzt. Diese verhindert unabhängig von Fahrspuren oder erkannten Hindernissen, dass es im Fahrbetrieb zu einem ZusammenstoSS mit anderen Gegenständen oder der Wand kommt. Hierzu werden die Daten des vorderen Ultraschallsensors ausgewertet, der einen ungefähren Abstandswert zum nächstgelegenen Gegenstand liefert.

Um aus dem fehlerbehafteten Signal verwertbare Daten zu erhalten werden die Werte des Sensors geglättet. Dies geschieht indem die letzten 20 Sensorwerte gemittelt werden. Liegt der Durchschnitt unter einem Schwellwert, wird der Motor angehalten. Als praxistauglicher Wert hat sich 0.35 erwiesen, also ein Abstand von 35 Zentimetern.

Ein besonderes Problem ergibt sich aus der Tatsache, dass die Werte des Sensors in unregelmäSSigen Abständen kurzzeitig auf den Wert Null springen. Dies entspräche einem Gegenstand direkt vor dem Sensor, was schon alleine aufgrund der Fahrzeuggeometrie nicht plausibel ist. Diese Werte müssen daher von der Auswertung ausgenommen werden.

Codeausschnitt: collision_protection()

Bild des Ultraschallsensors

4.5 Regelungskonzept und Spurhaltung

Nach umfangreichen Tests zu Beginn des Projekts wurde ein strukturvariabler Regelungsansatz mit PD-Reglern gewählt. Eine strukturvariable Regelung lässt sich vor Allem dann sinnvoll einsetzen, wenn zwischen wenigen bekannten Arbeitspunkten umgeschaltet werden muss. Die Rennstrecke des Projektseminars Echtzeitsysteme lässt sich Kurven- und Geradenabschnitte aufteilen. Aufgrund der hohen Nichtlinearität der Lenkungsmechanik (insbesondere Lose, Haftreibung, Gleitreibung) lässt sich mit einem

einzigen PD-Regler kein zufriedenstellendes Regelverhalten in allen Fahrsituationen erzielen. Eine stationäre Genauigkeit ist in diesem Fall zur Spurhaltung im Übrigen nicht erforderlich, solange die bleibende Regelabweichung in der Praxis so klein ist, dass alle Fahrsituationen der Rennstrecke ohne Linienüberschreitung gefahren werden können. Mit einem gut eingestellten PD-Regler lässt sich diese Anforderung erfüllen.

Die Bildverarbeitung liefert die Positionsdaten aller Markierungslinien, die im Bildausschnitt der Kamera erfasst werden. Da die Position der Kamera auf dem Fahrzeug sich nicht ändert, kann ein fester Sollwert für den Abstand zu diesen Linien vorgegeben werden. Es muss zur Berechnung der Regelabweichung lediglich bekannt sein, ob das Fahrzeug sich an der rechten oder an der linken Fahrbahnbegrenzung orientieren soll. Näheres hierzu in Abschnitt 4.7.

4.6 Implementierung des PD-Reglers

Die Berechnung der Lenkwinkelregelung erfolgt in mehreren Schritten. Zunächst wird aus den situationsabhängigen Reglerparametern sowie FührungsgröSSe und aktueller Position eine StellgröSSe berechnet. Der differentielle Anteil des PD-Reglers greift hierbei auf den aktuellen und vorherigen Wert der Regelabweichung, sowie die verstrichene Zeitdauer seit der letzten Berechnung zu.

Codeausschnitt: PD Regler

AnschlieSSend erfolgt eine Begrenzung der StellgröSSe. Die Hardware erlaubt Lenkwinkel im Bereich -800....+800. Um Beschädigungen zu vermeiden wird der maximale Lenkwinkel softwareseitig auf -700....+700 begrenzt. Bei höheren Werten kommt es zu einem Blockieren des Lenkgetriebes.

Um ein ruhiges Lenkverhalten zu erreichen werden die Ausgangswerte des Reglers zum Schluss geglättet. Dazu wird ein gewichteter Mittelwert der letzten StellgröSSen gebildet. Die Gewichtung erfolgt in Abhängigkeit des zeitlichen Verlaufs. Dabei hat der zuletzt berechnete Wert das gröSSte Gewicht, ein 0,3 Sekunden zurückliegender Wert hingegen nur das halbe Gewicht. Der gewichtete Mittelwert erzeugt ein ruhiges Regelverhalten, ohne jedoch die Regelung zu stark zu verzögern.

Codeausschnitt: Regler Glättung

4.7 Implementierung verschiedener Fahrsituationen

Das Konzept sieht die Aufteilung der zu fahrenden Strecke in vier wiederkehrende Fahrsituationen vor:

Aufzählung: Geradeausfahrt, Kurvenfahrt, Übergang von Geraden- zu Kurvenfahrt, Übergang von Kurven- zu Geradenfahrt

Je nach Situation wird zwischen einem Regler für Geradeausfahrt und einem Regler für Kurvenfahrt umgeschaltet. Die Übergangszustände dienen lediglich der stufenweisen Geschwindigkeitsanpassung beim Wechsel zwischen den beiden Streckenabschnitten. Die Umschaltung zwischen den verschiedenen Reglern erfolgt in Abhängigkeit vom gewählten Fahrmodus (siehe Abschnitt 4.9).

Der Regler für die Geradeausfahrt zeichnet sich durch aus, während der Regler für die Kurvenabschnitte
. (.....)
(.....)

4.8 Einleitung eines Spurwechsels

Die Implementierung eines Spurwechsels eröffnet vielfältige Möglichkeiten. Während ein Spurwechsel auf geraden Abschnitten der Strecke unproblematisch ist, kann es im Zusammenhang mit Kurven zu Problemen kommen. Wird eine Kurve auf der äuSSeren Spur durchfahren, so ist während der Kurvenfahrt grundsätzlich kein Wechsel auf die innere Spur möglich. Die konstruktive Begrenzung des Lenkwinkels wird durch die Kurvenfahrt schon komplett ausgenutzt, sodass kein Spielraum für ein Ausweichen nach innen mehr bleibt. Weniger problematisch ist das Ausweichen von der inneren auf die äuSSere Spur während einer Kurvenfahrt.

Auf der Rennstrecke des Projektseminars gibt es nur zwei Varianten eines Spurwechsels: von der rechten auf die linke Spur, und von der linken auf die rechte Spur. Grundsätzlich können diese Wechsel vorgenommen werden, indem der Regler-Sollwert auf den jeweils anderen Fahrbahnrand gesetzt wird. Dazu muss jedoch zunächst sichergestellt werden, dass sich überhaupt beide Fahrbahnmarkierungen im Sichtfeld der Kamera befinden. Daher wird (....)

4.9 Fahrmodi und Wettbewerb

Im Rahmen dieses Projektseminars sind zwei Aufgaben zu erfüllen: das Absolvieren einer kompletten Runde in möglichst geringer Zeit, und die Hinderniserkennung mit Spurwechsel bei möglichst wenigen Linienübertritten. Um beide Aufgaben möglichst gut zu lösen werden zwei verschiedene Fahrmodi implementiert, die sich durch die Fahrgeschwindigkeit und die Kriterien zur Wahl der Fahrspur unterscheiden.

4.9.1 Fahrmodus 1: Rundenzeit

tbd

Im ersten Fahrmodus liegt die Herausforderung in der Minimierung der Rundenzeiten. In diesem Modus wird ohne Hindernisse auf der Strecke gefahren, daher ist die Hinderniserkennung abgeschaltet.

4.9.2 Fahrmodus 2: Hinderniserkennung und Spurwechsel

In diesem Modus spielt die Fahrgeschwindigkeit eine untergeordnete Rolle, weshalb sie auf einen konstanten und relativ niedrigen Wert gesetzt wird. Zunächst orientiert sich das Fahrzeug am rechten oder linken Fahrbahnrand und folgt der Spur. Solange sich kein Hindernis auf der Strecke befindet, wird dieser Zustand beibehalten.

Ein Spurwechsel setzt voraus, dass dem Fahrzeug die eigene Position auf der Fahrbahn bekannt ist. Da dem Fahrzeug vorgegeben wird auf welcher Fahrspur es fahren soll, und die Regler ein schnelles Regelverhalten besitzen, kann im Allgemeinen davon ausgegangen werden, dass sich das Fahrzeug auf der vorgegebenen Fahrspur befindet.

Sobald ein Hindernis in das Sichtfeld der Kamera eintritt liefert die Bildverarbeitung dessen Koordinaten. Da auch die Koordinaten der beiden Fahrspuren bekannt sind, kann aus den Daten ermittelt werden, ob sich das Hindernis auf der aktuellen Fahrspur des Autos befindet. Befindet sich das Hindernis auf der eigenen Fahrspur, wird rechtzeitig ein Spurwechsel veranlasst. Andernfalls wird das Hindernis ignoriert und die Fahrt fortgesetzt.

Der Spurwechsel wird durch die Erkennung eines Hindernisses eingeleitet. Die Bildverarbeitung liefert dabei die Positionen des rechten und linken Randes eines Hindernisses, sodass neben der Position auch dessen Breite bekannt ist (siehe 4.3). Für den Spurwechsel entscheidend ist die Frage, ob sich das erkannte Hindernis überhaupt auf der gleichen Fahrspur wie das Fahrzeug befindet. Ist das nicht der Fall, stellt es kein Hindernis dar und kann ignoriert werden.

Codeausschnitt: Hinderniserkennung

5 Probleme

5.1 Kinect - unnatürliche Farben

Die Kamera der Kinect hat eine unnatürliche Farbdarstellung. Es sind sind starke Unterschiede zwischen Kamerabild und der Wirklichkeit zur erkennen.

Zeige ein Bild der Kinect und ein Bild der Webcam

Die Farben der Kinect sind übersaturiert und haben teilweise nicht einmal den richtigen Farbton.

5.2 Kinect - kleines Sichtfeld

Das Sichtfeld der Kinect Kamera fällt relativ klein aus. Das hat auch damit zu tun, dass die Kinect in einem Winkel auf dem Fahrzeug angebracht ist, mit dem das Sichtfeld erst circa 30cm vom Fahrzeug entfernt beginnt. Dadurch kann der Abstand des Fahrzeuges zur Fahrbahnbegrenzung zum aktuellen Zeitpunkt nicht ohne weiteres bestimmt werden. Zudem ist die Kamera in der Kinect am rechten Rand platziert. Dadurch ist die linke Fahrbahnbegrenzung häufig nicht im Sichtfeld. Dieser Effekt wird noch durch die Transformation zum Bird-Eye View verstärkt.

Vllt Bild von Webcam und Kinect zum Vergleich der Sichtfelder

Um eine robuste Fahrbahnerkennung zu implemetieren, sind ordentliche Ausgangsdaten von groSSer Wichtigkeit. In unserem Fall benötigen wir eine natürliche Farbdarstellung, ein ausreichend groSSes Sichtfeld das möglichst nah am Fahrzeug beginnt und eine flüssige Bildrate. Alle diese Anforderungen werden von der Webcam erfüllt. Die Webcam kann zudem sehr flexibel auf dem Fahrzeug angebracht werden.
Die Halterung der Webcam verursacht allerdings ein weiteres Problem. Die Webcam sakt durch Vibrationen die beim Fahren entstehen nach einiger Zeit etwas ab. Das heiSSt die Webcam zeigt mehr in Richtung Boden. Das hat merkliche Auswirkungen auf die Transformation zum Bird-Eye View. Wenn der Bird-Eye View nicht mehr korrekt berechnet wird, funktioniert die Erkennung der Fahrsituation nicht mehr so gut. Genauer wird nicht mehr erkannt, wann sich das Fahrzeug auf einer Geraden befindet. Da die Regelung abhängig von der Fahrsituation ist, kann das Fahrverhalten schlechter werden.

5.3 Webcam verändert Belichtung zur Laufzeit

Die Webcam passt ihre Belichtung an die aktuellen Lichtverhältnisse zur Laufzeit an. Das führt leider dazu, dass selbst kleine Veränderungen der Lichtverhältnisse die Belichtung ändern, wie zum Beispiel Lichtspiegelungen auf dem Boden. Wenn sich die Belichtung ändert, ändern sich auch schlagartig die Farbwerte. Um keine komplizierte Anpassung der Filterparameter während der Laufzeit vornehmen zu müssen, muss die Belichtung konstant gehalten werden. Das ist bei dieser Webcam nicht in Software machbar. Wir haben einen pragmatischen Ansatz gewählt um dieses Problem zu lösen. Wir blenden die Webcam mit einer LED Leiste. Die LED Leiste wird über das 12V Boardnetz gespeiSSt. Damit nimmt die

Webcam die Umgebung immer sehr hell wahr. Leichte Veränderungen der tatsächlichen Lichtverhältnisse werden durch das Blenden einfach herausgefiltert. Das Ergebnis ist eine konstante Belichtung und somit auch konstante Farbwerte.

5.4 Fahrwerk

Die Räder kollidieren bei zu hohem Lenkwinkel mit dem Gehäuse. Deswegen begrenzen wir den maximalen Lenkwinkel auf einen ausgemessenen Wert.
Die Lenkmechanik des Fahrzeugs ist aus Plastik und nicht sonderlich genau gefertigt. Das hat zur Folge, dass die Lenkung ein merkliches Spiel hat. Für uns bedeutet das, dass kleine Veränderungen am Lenkwinkel keine realen Auswirken haben. AuSSerdem liegt relativ viel Last auf der Vorderachse, weil der Laptop Akku und die Kinect vorne im Fahrzeug platziert sind. Dadurch wird die Lenkmechanik noch zusätzlich belastet. Bei höheren Geschwindigkeiten sollte sich dieser Effekt allerdings abschwächen. Durch diese Effekte können wir uns nicht darauf verlassen, dass nach dem Einstellen eines Lenkwinkels, dieser auch tatsächlich anliegt. In der Praxis fahren wir allerdings immer mindestens so schnell, dass dieser Effekt keine allzu groSSen Auswirkungen hat.
Ein weiteres Problem besteht darin, dass der Lenkwert der an die UC_Bridge gesendet wird, sich nicht linear in einen Lenkwinkel umrechnen lässt. Die Beziehung zwischen Lenkwert und Lenkwinkel auszumessen gestaltet sich schwierig. Da diese Beziehung stark von der Geschwindigkeit und der Last auf die Lenkmechanik abhängt. Wenn der tatsächliche Lenkwinkel durch passende Sensoren regelmäSSig gemessen werden würde, könnte man eine Regelung für den Lenkwinkel konzipieren. Wir haben uns um dieses Problem nicht weiter gekümmert und haben uns stattdessen darauf konzentriert eine möglichst robuste Regelung zur Fahrspurhaltung zu implementieren. Um die Lenkmechanik nicht unnötig zu belasten haben wir eine möglichst ruhige Regelung vorgenommen.

6 Literaturverzeichnis

[1] Fachgebiet Echtzeitsysteme TU Damrstadt. Einführung in ROS und die PSES-Plattform, 2017.