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

0.1 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

0.2 Grundlagen

0.2.1 Modellauto

evtl. Hardwaregrundlagen

0.2.2 OpenCV

0.2.3 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.

0.2.4 PSES Packages

1 Organisation

In diesem Abschnitt wird das Projektmanagment des Teams beschrieben, das zum erfolgreichen Abschluss unseres Projektes beigetragen hat. Bei einem Team mit fünf Personen ist es wichtig Maßnahmen zur Dateiverwaltung und Organisation zu treffen, um eine gute Zusammenarbeit anzusteuern.

1.1 Gruppentreffen und Organisation

Gruppentreffen

Wir entschieden uns bei unserem ersten Teamtreffen für eine Versionsverwaltung und eine Organisationsplattform und richteten diese zusammen ein.

Im zwei Wochen Takt traf sich vorzugsweise das gesamte Team mit einem Betreuer des Projektes. Mit ihm besprachen wir den aktuellen Stand unseres Projektes und gegebenfalls Probleme im Team. Diese Treffen waren organisatorischen Zwecken gewittmet, es wurden keine Fragen direkt zur Implementierung geklärt. Als das Projekt grundlegend funktionierte, wurden Anregungen zur Erweiterung der Aufgabenstellung gegeben.

Zur Besprechung von Regelungs- und Implementierungsfragen, gab es annähernd jede zweite Wochen ein Regelungstechniktreffen, bei dem sich zwei Teammitglieder jeder Gruppe einfunden. Es wurde der Stand der jeweiligen Gruppen untereinander ausgetauscht und Fragen diskutiert. Für weitere Fragen und Anregungen stand dort ein Betreuer der Regelungstechnik zur Verfügung.

Zusätzlich zu den Beratungsgesprächen legten wir feste wöchentliche Treffen mit allen Gruppenmitgliedern fest. Jeder berichtete von seinem Vorgehen der vergangenen Woche, so dass alle im Team den Überblick über den aktuellen Stand des Projektes hatten. Daraufhin wurden Ideen und Anregungen zusammengetragen und die Planung für die nächste Woche erarbeitet. Mit Hilfe der im Folgenden noch erwähnten Organisationsplattform Trello wurden die Aufgaben erfasst und den interessierten Gruppenmitgliedern zugeordnet.

Team

Wir entschieden die Zeitplanung dinamisch zu halten, da es schwer einschätzbar ist, wie viel Zeit eine einzelne Aufgabe in Anspruch nimmt. Erfahrungen aus anderen Projekten haben gezeigt, dass dieses Vorgehen sinnvoll ist. Wir setzten uns als Ziel, dass nach 2/3 der Projektzeit eine funktionierende Version vorhanden ist, um genügend Zeit für die Optimierung und Dokumentation einzuplanen und gegebenenfalls einen ausreichenden Zeitpuffer zur Verfügung zu haben. Jede Woche bei dem Gruppentreffen wurde festgelegt, wer sich welcher neuen Aufgabe annimmt und wir schätzten erneut ab, wie viel Zeit eine angefangegne Aufgabe noch benötigen wird.

Nachdem sich alle in die Dokumentation eingelesen hatten, erarbeiteten wir erste Grundlagen zusammen, damit jeder ein Grundwissen über das Modellauto hatte. Es bildeten sich Aufgabengruppen für Organisation, Regelung und Bildverarbeitung, diese wurden weitgehendst parallel behandelt. Die Verantwortung für diese Teilgruppen wurde auf die Mitwirkenden aufgeteilt. Jedes Gruppenmitglied konnte bei jeder

Aufgabengruppe helfen, allerdings ist es wichtig Verantwortliche festzulegen, um den Überblick zu bewahren und Zielführend zu agieren. Als die Tests der Bildverarbeitung und Regelung die aufgestellten Bedingungen erfüllten, schlossen wir diese Aufgabenbereiche zusammen und erlangten dadurch das Fahren durch den Rundkurs. Neben der Optimierung des Rundkurses wendete sich eine Subgruppe dem zweiten Thema, Hinderniserkennung mit Spurwechsel, zu.

Es gab viele Aufgaben, die direkt am Fahrzeug getestet werden mussten. Da das gleichzeitige Testen mehrerer Aufgaben meist nicht möglich war, unterstützte man beim Zusammenkommen die anderen Teammitgliedern über die Aufgabenverteilung hinaus. Jeder teilte sich seine Zeit selbstständig jede Woche ein. Wir kommunizierten, welche Aufgabe wann bearbeitet wird, damit Interessengruppen zusammenfinden konnten.

1.2 Aufgabenverwaltung

1.3 Versionsverwaltung

2 Implementierung

2.1 Bildverarbeitung und Erkennung von Fahrspuren

Fabian

Zur Bilderzeugung konnte zwischen zwei Kamerasystemen wählen werden: Einer Kinect Kamera sowie einer Weitwinkel Webcam. Die Entscheidung fiel auf die Weitwinkel Webcam, da sich diese aufgrund ihrer Weitwinkel- und Farbdarstellungseigenschaft und den in Kapitel 3 genannten Problemen der Kinect Kamera sich als besser geeignet herausstellte.

Die mittels OpenCV eingelesenen Kamerabilder werden durch die Natur der Kameraposition in Zentralperspektive aufgezeichnet. Diese Art der Projektion eignet sich aufgrund des vorhandenen Fluchtpunktes nicht für die benötigte Bildverarbeitung, weshalb eine Transformation in Vogelperspektive durchgeführt wird, welche die Problematik des Fluchtpunktes beseitigt.

Zur Erkennung der Fahrspuren wurden die Farbwerte der grünen äußeren Fahrbahnmarkierungen ermittelt. Mithilfe dieser Farbwerte kann ein Graustufenbild erzeugt werden, welches die Übereinstimmung mit dem vorgegebenen Farbwert repräsentiert. Je mehr der Farbpixel des Originalbildes mit dem verglichenen Farbwert übereinstimmt, desto heller erscheint der Pixel an seiner Stelle im Graustufenbild. Um die Orientierung in diesem sicherzustellen werden drei Marker in äquidistanter Höhe auf die rechte und linke Linie gesetzt (siehe Abb. 2.1). Der Algorithmus zur Positionierung der Marker sucht dabei zeilenweise im Bild nach aufeinanderfolgende helle Pixel und addiert die Helligkeit dieser Pixel auf. Übersteigt diese Summe einen Schwellwert, so wird diese Stelle mit einem Marker versehen.

Bild von Fahrspur mit roten und grünen Punkten

2.2 Erkennung von Kurven und Geraden

Um das Fahrverhalten besser anpassen zu können, fiel die Entscheidung darauf Fahrmodi zu erstellen, welche unterschiedliche Reglerparameter besitzen. Um diese Fahrzustände korrekt erkennen zu können ist es von Nöten zwischen Geraden, Kurven, Kurvenein- und Kurvenausfahrten zu unterscheiden.

Um eine Kurve zu erkennen werden die drei gesetzten Marker betrachtet. Da sich diese in gleichem vertikalen Abstand zu einander befinden wird lediglich der horizontale Abstand betrachtet. Zuerst wird hierfür der vertikale Abstand zwischen dem Unteren sowie dem Mittleren Marker ermittelt. Dieses Verfahren wird mit dem Oberen und dem Mittleren wiederholt. Übersteigt die Differenz der beiden ermittelten Abstandswerte einen Schwellwert, so liegt eine Kurve vor

Um nun auch Kurvenein- und Kurvenausfahrten erkennen zu können wurde ein Puffer implementiert, welches aus vorherigem und aktuellem Kurvenerkennungswert ermitteln kann in welcher Umgebungssituation sich das Fahrzeug befindet. Des weiteren dient der Puffer als eine Hysterese welche ein Flackern der Fahrzustände vermeidet.

Bild von Kurve aus AUDO Sicht

2.3 Erkennung von Hindernissen

Wie auch die Fahrspurerkennung basiert die Hinderniserkennung auf der Filterung der Kamerabilder auf eine spezielle Farbe. Aus diesem Grund wurden die Hinderniswürfel, um sie vom Rest der Kulisse unterscheiden zu können mit orangenem Klebeband versehen. Um sicher zu gehen, dass nicht fälschlicherweise ein Hindernis erkannt wird, ein ähnlicher Algorithmus angewandt er zuvor in Kapitel 2.1 beschrieben wurde. Des weiteren wird ein Hindernis nur als solches erkannt, wenn zwei vertikale Linien nebeneinander erkannt werden, wie es bei den Hinderniswürfeln der Fall ist.

Fabian

2.4 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

2.5 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 2.7.

2.6 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

2.7 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 2.9).

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

(.....)

(.....)

2.8 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 (....)

2.9 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.

2.9.1 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.

2.9.2 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.

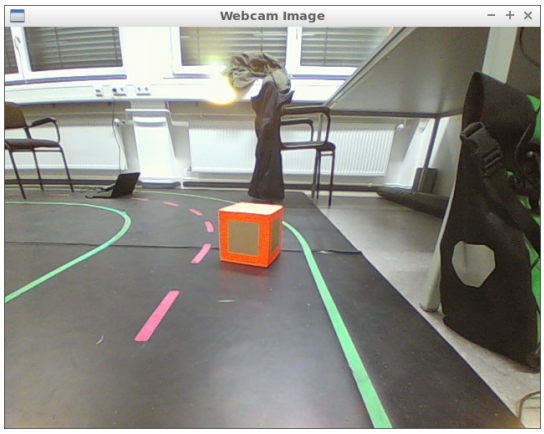
Der Spurwechsel wird durch die Erkennung eines Hindernisses eingeleitet. Die Bildverarbeitung liefert dabei die Positionen des rechten und linken Randes eines Hindernisses, sodass neben der Position auch dessen Breite bekannt ist (siehe 2.3). Für den Spurwechsel entscheidend ist die Frage, ob sich das 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

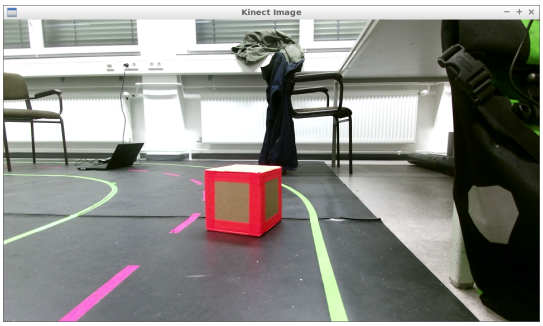
3 Probleme

3.1 Kinect - unnatürliche Farben

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



(a) Webcam Original



(b) Kinect Original

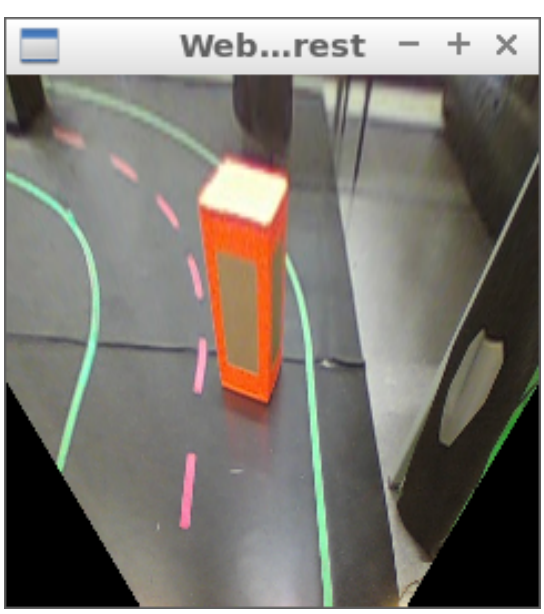
Abbildung 3.1: Farbdarstellung von Webcam und Kinect

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

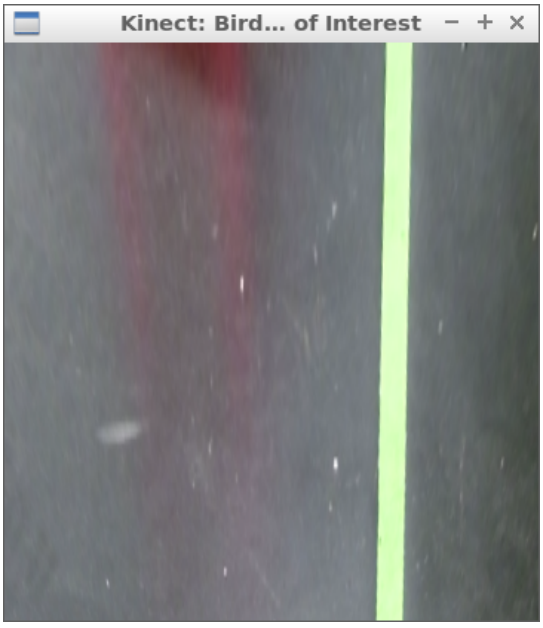
3.2 Kinect - kleines Sichtfeld

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.

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.



(a) Webcam Bird-Eye View und Bildausschnitt



(b) Kinect Bird-Eye View und Bildausschnitt

Abbildung 3.2: Sichtfeld von Webcam und Kinect

3.3 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.

3.4 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.

4 Literaturverzeichnis

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