Das Projekt cITicar

Autonomes Fahren als begleitende Lehr- und Lernplattform – auch in der Pandemie

Jakob Czekansky¹, Moritz Schauer¹, Florian von Zabiensky¹, Diethelm Bienhaus¹

Zusammenfassung

Das Projekt cITicar bietet ein breites Spektrum für das Lehrangebot im Bereich der (Technischen) Informatik und der Ingenieur-Informatik. Aus dem Themengebiet der Eingebetteten Systeme kann sowohl die Prototypenentwicklung als auch die Mikroprozessorprogrammierung und das Arbeiten mit Echtzeitsystemen am cITicar erlernt werden. Typische Aufgaben aus der Regelungstechnik können am Fahrzeug durchgeführt und verschiedene Probleme aus der Längs- und der Querregelung von den Studierenden gelöst werden. Zusätzlich bietet das cITicar Potential für Aufgaben aus dem Bereich der Computer Vision und des Machine Learnings: Fahrbahnranddetektion, Schilderkennung oder Fahrbahnsegmentierung sind nur wenige Beispiele für Lehraufgaben am cITicar. Sowohl die Lehrmodule als auch die Projektarbeit am cITicar wurden durch die Pandemie vor neue Herausforderungen gestellt. Um die Onlinelehre im Projekt zu verbessern, wurden verschiedene Werkzeuge und Techniken in der Lehre integriert. In dieser Arbeit werden diese Werkzeuge und Techniken, sowie die damit gewonnenen Erfahrungen dargelegt.

Stichwörter

Autonomes Fahren, Mobile Robotik, Simulation, Lehrkonzept, Projektarbeit, Eingebettete Systeme, Testing, Remote-Programming, Continuous Integration, Carolo-Cup

1 Einleitung

„Jede dritte Fahrt erfolgt im Jahr 2035 mit einem autonomen Flottenfahrzeug“ – so prognostiziert zumindest das Wirtschaftsunternehmen Deloitte die nahe Zukunft des autonomen Fahrens [1]. Auch an dem weltweit starken Zuwachs an Patenten mit Bezug zum autonomen Fahren [2][3] und dem global wachsenden Bestand an Elektrofahrzeugen [4] lässt sich dieser Trend erkennen. Um die Studierenden der Technischen Hochschule Mittelhessen (THM) auf dieses weitreichende Themengebiet der mobilen Robotik vorzubereiten, wurde das studienbegleitende Projekt cITicar (Abbildung 1) ins Leben gerufen. Dabei handelt es sich um ein autonom fahrendes Modellfahrzeug im Maßstab 1:10, welches in regelmäßigen Abständen an verschiedenen internen und externen Wettbewerben im Bereich des autonomen Fahrens teilnimmt. 2020 konnte das cITicar das erste Mal am Carolo-Cup der Technischen Universität Braunschweig teilnehmen und belegte dort den sechsten von vierzehn Plätzen im Basic Cup. Bei dem Projekt cITicar handelt es sich um ein rein studentisches Lehr- und Lernprojekt, an dem die Studierenden verschiedener Studiengänge der THM teilhaben können. Da das Projekt an der THM am Fachbereich für Mathematik, Naturwissenschaften und Informatik (MNI) angesiedelt ist, liegt der Fokus bei der Entwicklung besonders auf der Hardware- und Softwareentwicklung. Neben freiwilligen Arbeiten der Studierendenschaft am cITicar wird die Weiterentwicklung des Fahrzeugs durch

¹ Institut für Technik und Informatik, Technische Hochschule Mittelhessen, Gießen

Bild 1: Das cITicar 2020

verschiedene Wahlpflichtmodule und das Angebot von Abschlussarbeiten im Kontext des Modellfahrzeugs gefördert. Im weiteren Verlauf dieser Arbeit soll zunächst in groben Zügen der zugrundeliegende Carolo-Cup vorgestellt werden, um anschließend das Projekt cITicar in dessen Kontext zu beleuchten. Während der Pandemie sollte das Projekt möglichst effizient weiterlaufen, weshalb die Onlinelehre im Kontext des cITicars so konzipiert wurde, dass die Studierenden von Zuhause aus das autonome Fahrzeug weiterentwickeln können. Hierzu wurden verschiedene Maßnahmen ergriffen, welche in Abschnitt 4 beschrieben werden und als Inspiration für ähnliche Projektarbeiten dienen können.

2 Der Carolo-Cup

Der Carolo-Cup¹ ist ein internationaler Hochschulwettbewerb, der seit 2008 jährlich im Februar von der Technischen Universität Carolo-Wilhelmina zu Braunschweig angeboten wird [5]. Der Wettbewerb inkludiert verschiedene statische und dynamische Disziplinen, in denen Teams aus internationalen Hochschulen und Universitäten mit autonom fahrenden Modellfahrzeugen im Maßstab 1:10 gegeneinander antreten können. Im Jahr 2008 wurde der Wettbewerb initial von sechs teilnehmenden Teams besucht, wohingegen 2020 bereits 22 Teams am Carolo-Cup partizipiert haben.

Der Carolo-Cup teilt sich intern in zwei verschiedene Schwierigkeitsklassen auf: den Basic-Cup und den Master-Cup. Im Basic-Cup wird unter vereinfachten Bedingungen konkurriert, während im Master-Cup zusätzliche Schwierigkeiten beim Rundkurs eingebaut werden. Der Basic-Cup richtet sich in erster Linie an einsteigende Teams, mit wenig Erfahrung, wohingegen der Master-Cup eine Herausforderung für die Teams bieten soll, die schon seit einigen Jahren am Wettbewerb teilhaben. In beiden Klassen wird der Wettbewerb jeweils in unterschiedlichen Disziplinen bewertet: Bei den sogenannten Statischen Disziplinen muss das Gesamtkonzept der einzelnen Fahrzeuge in einem Fachvortrag vor einem Bewertungskomitee aus Lehre und Industrie präsentiert werden. Zusätzlich bewertet das Komitee verschiedene technische, wirtschaftliche und organisatorische Aspekte im Bezug auf das vorgestellte Fahrzeug [6]. In der Dynamischen Disziplin müssen die Teams anschließend jeweils für zwei Minuten einen bestimmten Rundkurs bezwingen. Beim ersten Durchlauf, im Modus „Freie Fahrt und Einparken“, kommt es darauf an, auf dem Rundkurs so viele Meter wie möglich zurückzulegen und dabei möglichst wenige Fehler zu machen. Zusätzlich kann bei jedem Rundendurchgang einmal vorwärts oder seitlich rückwärts eingeparkt werden, um die zurückgelegten Meter mit einem Multiplikator zu versehen. Im zweiten Durchlauf, dem „Hindernisparkours“ wird die Fahrbahn an verschiedenen

¹ <https://www.tu-braunschweig.de/carolo-cup/>

Stellen durch feste, statische Hindernisse und durch bewegliche, fahrende Hindernisse blockiert. Das autonome Fahrzeug muss bei diesem Kurs an passenden Stellen überholen und Vorfahrtsregeln an Kreuzungen beachten. Die Punkte der Statischen und der Dynamischen Disziplin werden schlussendlich verrechnet und die besten Teams werden mit Preisgeldern belohnt. 2020 wurde der erste Platz im Basic-Cup mit 2000€ dotiert, der zweite Platz mit 1000€ und der dritte Platz mit 500€. Zusätzlich muss das erstplatzierte Team des Basic-Cups bei erneuter Teilnahme im Folgejahr im Master-Cup teilnehmen [7].

3 Das Projekt cITicar

Das Projekt cITicar² wurde im Sommersemester 2018 am Fachbereich MNI der THM gegründet, mit dem Ziel, ein vielseitig integrierbares Projekt für die Studierenden der Informatik und der Ingenieur-Informatik zu schaffen. Ähnliche Projekte gibt es in einigen Universitäten und Hochschulen Deutschlands, wie beispielsweise im Jahr 2018 von der Hochschule RheinMain publiziert wurde [8]. An der Hochschule RheinMain wurde das Autonome Fahren am Beispiel des Carolo-Cups im Rahmen einer Projektveranstaltung *Wahlprojekt* in die Lehre integriert [8]. Das cITicar der THM soll die Studierendenschaft jedoch über viele Semester ihres Studiums bis hin zu einer potentiellen Abschlussarbeit begleiten und in unterschiedlichen Lehrmodulen als Demonstrations- und Arbeitsplattform dienen. Hierzu wurde beim ersten studentischen Entwurf des Fahrzeugs darauf geachtet, dass eine möglichst breite Abdeckung der typischen Technologien aus dem Bereich der (technischen) Informatik vorliegt. Zusätzlich wurde das cITicar einem realen Elektrofahrzeug nachempfunden, sodass explizit Bussysteme, wie ein Controller Area Network (CAN) Bus, im Fahrzeug verbaut wurden [7].

3.1 Hardware- und Softwarekonzept des cITicars

Das cITicar basiert zurzeit auf dem 1:10 RC-Chassi „Deathwatcher EVO 2.0“ des Modellbauherstellers Reely. Neben Modifikationen an der Mechanik wurde die komplette Steuerungselektronik des Modellsatzes ersetzt, um das ferngesteuerte Modell auf den autonomen Betrieb umzurüsten. Dabei wurde sowohl die Software als auch die zugehörige Hardware in verschiedene Ebenen eingeteilt. Grob lässt sich der Aufbau in ein Eingebettetes System und einen Mini-Computer (Intel Next Unit of Computing (NUC)) einteilen (siehe Abbildung 2). Dabei dient der NUC als Ebene der Zentralen Intelligenz des Fahrzeugs. Hier wird auf der einen Seite die gesamte Verhaltenslogik des Fahrzeugs implementiert. Auf der anderen Seite wurde hier auch die rechenintensive Bildverarbeitung, welche zur Navigation des Fahrzeugs genutzt wird, umgesetzt. Die Kommunikation der beiden Softwareteile untereinander aber auch mit den anderen Komponenten wird mithilfe des ROS-Frameworks umgesetzt [7].

Neben der übergeordneten Intelligenz müssen auch die anderen Aktoren und Sensoren des Fahrzeugs angesteuert werden. Hierfür wurde eine eigene Steuerplatine entwickelt. Auf ihr befinden sich neben den beiden Cortex M4 Mikrocontrollern diverse Anschlüsse für die unterschiedlichen Komponenten sowie die zum Einsatz kommenden Bus-Systeme. Dabei sind die beiden Microcontroller Units (MCUs) in High- und Low-Level-Aufgaben eingeteilt. Der High-Level-Controller übernimmt hier erstens die Kommunikation mit der Zentralen Intelligenz und zweitens die Längs- und Querregelung des Fahrzeugs. Der Low-Level-Controller ist für die Kommunikation (Serial Peripheral Interface (SPI)- und Inter-Integrated Circuit (I2C)-Bus) und das Bereitstellen der Daten der verschiedenen Sensoren und Aktoren zuständig. Als Sensoren kommen bei dem Fahrzeug eine Inertiale Messeinheit sowie mehrere Time-of-Flight Sensoren zum Einsatz. Die Controller kommunizieren hierbei untereinander über den CAN-Bus des cITicars. Für die Ansteuerung des Fahrzeugmotors kommt ein weiteres Controller-Board zum Einsatz, welches ebenfalls an den CAN-Bus angeschlossen ist [7].

² <https://citicar.thm.de/>

Bild 2: Überblick über das Hardware- und Softwarekonzept des cITicars 2020

3.2 Einbindung des cITicars in die Lehre

Im Bereich der klassischen Informatik und der Ingenieur-Informatik finden sich viele Überschneidungen: Grundlagen zu Objektorientierter Programmierung oder Module zu Algorithmen und Datenstrukturen sind für die Studierenden beider Studiengänge wie vieles Basiswissen im Bereich der Informatik unumgänglich. Die Ingenieur-Informatik spezialisiert sich jedoch, ähnlich wie die bekanntere Technische Informatik, eher im Bereich der klassischen Ingenieurwissenschaften. Schwerpunkte werden im Studium besonders auf die Entwicklung von Eingebetteten Systemen, die systemnahe und die Mikroprozessorprogrammierung gesetzt. Im weiteren Verlauf des Studiums können sich die Studierenden anschließend weiter mit dem praktischen Entwurf von Elektronik und der Entwicklung von eingebetteten Prototypen beschäftigen. Im Kontext dieser Prototypenentwicklung hat sich über viele Jahre weiterhin der Einsatz von 3D-Druckern im Bereich der Ingenieur-Informatik an der THM etabliert, sodass die Studierenden lernen mit verschiedenen 3D-Druck-Techniken umzugehen und eigene CAD-Entwürfe anfertigen können. Darüber hinaus hat das Studium der Ingenieur-Informatik weitere Schwerpunkte im Bereich der Signalverarbeitung, der Computer Vision und des Maschinellen Lernens. Die Studierenden können n-dimensionale Signale sowohl mit klassischen Methoden, wie auch mit Deep-Learning-Verfahren be- und verarbeiten.

Um in der Ingenieur-Informatik den bestehenden Schwerpunkt im Bereich der mobilen Robotik zu vertiefen und zusätzlich eine Schnittmenge zwischen den bisher beschriebenen Themen zu schaffen, wurde das Projekt cITicar gegründet. Hier finden die bereits beschriebenen Kerngebiete der Ingenieur-Informatik gebündelt einen Platz. In Modulen wie *Praktische Elektronik und Prototypenentwicklung* können die Studierenden den kompletten Prozess vom Planen, über das Entwerfen bis hin zum Fertigen einer Platine am Beispiel einer eigenen Steuerplatine für das cITicar durchführen. Im *Praktikum Eingebettete Systeme* erlernen die Studierenden die saubere Programmierung von Mikrocontrollern und das Ansteuern verschiedener Peripherien in einem eingebetteten System, auf Basis der weit verbreiteten ARM Cortex-M Architektur. Im Bereich des Masterstudiums findet das cITicar Einsatz im Modul *Regelungstechnik*, in dem die Studierenden praktische Erfahrungen beim Längs- und Querregeln eines autonomen Fahrzeugs sammeln können. In *Echtzeitsysteme* finden die Masterstudierenden die Möglichkeit, ein eigenes Echtzeitbetriebssystem (RTOS) für das cITicar zu entwickeln und verschiedene Analysemethoden an dem Realprojekt anzuwenden.

Bild 3: Einbindung des cITicar in die Lehre der Informatik und der Ingenieur-Informatik

Neben den Einsatzgebieten in den beschriebenen, eher klassischen Lehrmodulen wurde zusätzlich das Curriculum des Bachelorstudiums um die Module *Computervision* und *Entwicklung eines autonomen Fahrzeugs*, und das Masterstudium um das Modul *Fortgeschrittene Konzepte des autonomen Fahrens* erweitert. Die Studierenden haben durch diese Wahlpflichtmodule die Möglichkeit, explizit in die Thematik der cITicar-Entwicklung einzusteigen. In diesem Kontext kann das notwendige Wissen im Bereich des Robot Operating System (ROS), ebenso wie die Grundlagen für die notwendige Bildverarbeitung am cITicar für Straßenrand-, Schild- oder Parklückenerkennung geschaffen und direkt angewendet werden. Auf diese Art können die Studierenden gezielt ihr favorisiertes Themengebiet am Projekt cITicar fokussieren und sich im weiteren Studium in diesem Bereich spezialisieren. Seit Beginn des Projekts konnten bereits weit über zehn Abschlussarbeiten (Bachelor und Master) im Kontext des cITicar durchgeführt werden. Die Themengebiete sind dabei weit gestreut: Abschlussarbeiten mit dem Schwerpunkt auf der *Modellierung der Hardware- und Software-Architektur eines autonomen Modellfahrzeugs in AADL*, die *Modellgetriebene Entwicklung eines Robotik-Systems für ein autonomes Fahrzeug* oder auch die *Modellierung von Hardware-Komponenten zur Integration in einer Simulationsumgebung verteilter, eingebetteter Systeme* bilden nur eine kleine Auswahl der erfolgreichen Thesen in dem Projekt.

4 Onlinelehre in der Pandemie

Eine der größten Herausforderungen, die es seit Beginn der Pandemie im Frühjahr 2020 in vielen Schulen, Hochschulen und Universitäten zu bewältigen gilt, ist die plötzliche Umstellung von Präsenzlehre auf Onlinelehre. Wie vorhergehend beschrieben, inkludiert das Projekt cITicar viele Arbeiten, die am cITicar direkt vor Ort unternommen werden müssen, weshalb das bestehende Lehrkonzept an vielen Punkten angepasst werden musste, um weiterhin erfolgreich zu funktionieren. Zunächst müssen hierzu die Probleme analysiert werden, welche durch die Onlinelehre entstanden sind, um anschließend adäquate Werkzeuge für die Lösung der Probleme zu finden und die Onlinelehre zu verbessern. Im Folgenden werden die entstandenen Probleme vorgestellt und im Anschluss die Lösungsansätze im Kontext des Projekts cITicar präsentiert.

Instant Messaging Kommunikationsplattform

Zu Beginn der Onlinelehre war die Infrastruktur der THM für digitale Kommunikation zwischen Dozierenden und Studierenden kaum ausgebaut, woraus eine mangelhafte Erreichbarkeit für beide Par-

teien resultierte. Mit der Einführung einer niedrigschwelligen Instant-Messaging-Plattform konnten gleich mehrere Probleme gelöst und eine entscheidende Grundlage für die Onlinelehre geschaffen werden. RocketChat³ ist eine Open-Source-Lösung, die zur Erhöhung des Datenschutzes auf hochschuleigenen Servern gehostet wird. Über dedizierte Kanäle wird hier gezielt themenspezifischer Austausch gefördert, während Direktnachrichten zwischen zwei oder mehreren Teilnehmenden den Rahmen für einen vertraulichen Austausch bieten. Da Nachrichten gespeichert bleiben, dient RocketChat darüber hinaus der einfachen Dokumentation von Diskussionen und Beschlüssen. Der Instant Messenger bewährte sich im Rahmen des cITicar-Projekts und wurde schließlich hochschulweit freigegeben. Rund 1600 aktiv Nutzende in über 1100 Chaträumen profitieren mittlerweile an der THM von dieser Lösung.

Virtuelle Arbeitsräume

Während sich RocketChat sowohl für synchrone als auch asynchrone schriftliche Kommunikation anbietet, musste auch für den verbalen Austausch in Lehrveranstaltungen und Besprechungen ein virtuelles Äquivalent gefunden werden. Auch hier war Datenschutz eine besondere Priorität, wodurch nicht auf übliche Enterprise-Lösungen zurückgegriffen werden sollte. Mit dem Open Source Tool BigBlueButton⁴, das ebenfalls auf hochschuleigenen Servern gehostet wird, können gemeinsame Termine (unabhängig vom Aufenthaltsort der Teilnehmenden) wahrgenommen werden. Weiterhin bieten virtuelle Arbeitsräume den Studierenden die Möglichkeit, sich jederzeit selbstorganisiert zu vernetzen und zusammenzuarbeiten. Auch dieses System war im initialen Projekt so erfolgreich, dass es im Anschluss in den Lehrkontext der gesamten THM übernommen wurde.

Containerisierung der Arbeitsumgebung mithilfe von Docker

Bereits während der Präsenzlehre war die Einrichtung der Entwicklungsumgebung für den ROS-basierten Teil des cITicar-Systems ein komplexer und langwieriger Prozess: Die Studierenden mussten für den Einsatz der verwendeten ROS Version das zugehörige Ubuntu in der passenden Version installieren. Lange Zeit basierte das cITicar auf ROS *Kinetic Kame*, welches wiederum ein Ubuntu in der Version 16.04 voraussetzt. Nach einer Umstellung des Codes auf ROS *Melodic Morenia* war es notwendig, dass die Studierenden künftig eine Ubuntu 18.04 Instanz auf ihrem System zur Verfügung hatten. Immer wieder kam das Feedback aus der Studierendenschaft, dass die Einrichtung der Entwicklungsumgebung zu komplex sei. Die Bereitschaft ein zusätzliches Ubuntu auf dem eigenen Gerät zu installieren, um an den zugehörigen Wahlpflichtmodulen teilzunehmen, sank. Diese Problematik bestand in der Präsenzlehre bereits, wurde jedoch durch die sich häufenden Problemstellen und die erschwerten Hilfsmöglichkeiten in der Onlinelehre weiter verstärkt. Um die Einrichtung der Entwicklungsumgebung zu erleichtern wurde daher eine containerisierte Arbeitsumgebung mithilfe von Docker geschaffen (siehe Abbildung 4).

Die Studierenden müssen sich lediglich ein einzelnes Repository klonen und anschließend plattformunabhängig⁵ die beinhaltete *docker-compose.yml* Datei ausführen. Auf diese Art werden automatisiert verschiedene Container für die Entwicklung mit dem cITicar erstellt: Die beiden Container ROS1 und ROS2 beinhalten jeweils die verschiedenen Softwareknoten des cITicars und sind über einen weiteren Container, ROS Bridge, miteinander verbunden. So können die Studierenden sowohl in ROS1 als auch in ROS2 Programme entwickeln und direkt im Projekt integrieren. Mithilfe des Gazebo Containers wird zusätzlich eine Simulationsumgebung auf Basis von Gazebo⁶ im System bereitgestellt. Hierzu wurde das komplette cITicar nachmodelliert, sodass eine virtuelle Version des Fahrzeugs von den ROS-Knoten in Gazebo angesteuert werden kann. Die Studierenden können mit dieser Konfiguration alle Entwicklungsschritte, die nicht im Eingebetteten System stattfinden, am eigenen Computer realisieren. Sobald einzelne Softwarekomponenten vollständig in der Simulation getestet wurden, kann anschließend ein Test an der Hochschule stattfinden, um das reale Verhalten des Fahrzeugs zu prüfen.

³ <https://de.rocket.chat/>

⁴ <https://bigbluebutton.org/>

⁵ Getestet wurden unter anderem Linux-Distributionen wie Linux Mint, Manjaro, Ubuntu, Debian und ElementaryOS in verschiedenen Versionen.

⁶ <https://gazebo.org/>

Bild 4: Struktur der containerisierten Arbeitsumgebung. Gestrichelte Pfeile stellen Dateisystem-Anbindungen dar. Durchgezogene Pfeile zeigen eine Kommunikation zwischen den Containern. Blaue Kästen sind Docker Container und der grüne Kasten stellt das Host-System dar. Runde Kästen dienen der Darstellung von verwendeten Softwarekomponenten. Farblose eckige Kästen stellen externe Peripherie dar.

Ausleihsystem für Hardwarekomponenten

Im Rahmen des Projektes aber auch generell in sehr vielen Bereichen der Ingenieur-Informatik ist das Arbeiten mit unterschiedlicher Hardware gefordert. Dies betrifft verschiedenste Sensoren, Aktoren oder auch Mikrocontroller-Evaluationsboards, um nur ein paar Beispiele zu nennen. Im Regelbetrieb wird diese Hardware in den verschiedenen Laboren bereitgestellt. Wegen der Pandemie sind die Studierenden jedoch darauf angewiesen, diese Hardware auch Zuhause benutzen zu können. Hierfür wurde das Open Source Asset Management Snipe-IT⁷ eingeführt. Dieses System wird über die hochschuleigene Serverinfrastruktur bereitgestellt. Es bietet die Möglichkeit, Hardware, Verbrauchsmaterialien und sogar Softwarelizenzen verwalten zu können. Hierbei können sowohl die einzelnen Komponenten verliehen, diese aber auch zu verleihbaren Sets zusammengestellt werden. Eine schnelle Identifizierung der Komponenten bei der Ausgabe beziehungsweise bei der Rücknahme lässt sich über vom System generierte QR-Code-Labels lösen. Ein weiterer Vorteil ist die Anbindung an das hochschuleigene LDAP-Verzeichnis. Dadurch müssen die Nutzungsdaten der einzelnen Studierenden und Mitarbeitenden nicht aufwendig gepflegt werden. Durch das digitale System war es auf der einen Seite möglich, die gesamten Hardwarekomponenten des Instituts zu inventarisieren und so auch stets über den Verbleib der Hardware informiert zu sein. Auf der anderen Seite hat es den gesamten Ablauf der Ausleihe stark vereinfacht, da nun eine transparente und dauerhafte Dokumentation existiert, bis hin dazu, dass die Studierenden automatisch an die terminierte Rückgabe der Hardware erinnert werden.

Remote Programmierung des Eingebetteten Systems

Um den Studierenden auch während der Pandemie die Arbeit mit physischer Hardware zu ermöglichen, wurde vor Ort im Labor der Hochschule ein Testaufbau erstellt. Dieser erlaubt es die daran angeschlossenen Mikrocontroller über einen Remote-Zugriff zu flashen und zu debuggen [9]. Grundsätzlich besteht der Testaufbau aus mehreren kleinen Komponenten. Das ganze System dahinter ist

⁷ <https://snipeitapp.com/>

in der nachfolgenden Abbildung 5 dargestellt. Die Steuerzentrale des Systems bildet auch hier ein NUC Mini-Computer, welcher als zentraler Server vor Ort arbeitet und über Programmieradapter in direkter Verbindung mit den Mikrocontrollern des cITicar steht. Auf dem NUC laufen drei OpenOCD⁸-Instanzen, die aus dem VPN erreichbar sind. OpenOCD steht für Open On Chip Debugger und nutzt die angeschlossenen Programmieradapter, um das Debug-Interface der verschiedenen Mikrocontroller zu bedienen. Zusätzlich setzt OpenOCD noch das Server Protokoll des GNU Debuggers (GDB) um. Somit kann man sich mittels einer GDB Instanz auf die OpenOCD Server verbinden und die Mikrocontroller hierüber flashen und debuggen [9]. Die Studierenden haben die Möglichkeit sich direkt aus dem Projekt der Entwicklungsumgebung mit dem gewünschten Mikrocontroller zu verbinden, ohne wirklich vor Ort sein zu müssen, und mit diesem zu arbeiten. Um die Server zu erreichen, müssen sich die Studierenden zunächst in das VPN der THM eingewählt haben. Da man beim Debuggen auch die Reaktion des Fahrzeugs bzw. der Hardware auf bestimmte Eingaben überprüfen muss, verfügt der Testaufbau über verschiedene Kameras, die rund um das Auto platziert wurden. Die Kameras sind ebenfalls per USB an die Steuerzentrale angeschlossen. Ein Kamera-Stream kann von den Studierenden über Motion⁹ abgerufen werden [9].

Bild 5: Aufbau des Systems zur Remote Programmierung mit GDB-Server, ST-Links, MCUs

Um gleichzeitig den Entwicklungsprozess zu verbessern, wurde für die verschiedenen Projekt-Repositorien in der Projektverwaltung je eine Continuous Integration Pipeline erstellt. Hierfür wurde ein GitLab Runner auf der Steuerzentrale eingerichtet. Dieser dient der Ausführung der Continuous Integration Pipeline bzw. der darin definierten Jobs. Die Pipeline dient der automatisierten Fehlersuche und dem Testen des Codes. Das Executable wird automatisiert gebaut und in einem anschließenden Job auf den jeweiligen Mikrocontroller geflasht. Um das Testen realer Hardware in den Continuous Integration Prozess mit aufzunehmen, wurde ein extra Testframework in Python entwickelt. Innerhalb des GitLab Runners wird in dem jeweiligen Job eine eigene GDB Instanz gestartet und anschließend wird das Framework sowie die Tests in den GDB geladen. Die Tests dienen der automatisierten Steuerung bzw. dem Debuggen des Mikrocontrollers über den GDB. Somit können die Tests verschiedene Debugging Szenarien durchlaufen und durch beispielsweise zusätzliche Peripherie auch ausgehende

⁸ <https://openocd.org/>

⁹ <https://motion-project.github.io/>

Nachrichten und Signale des Mikrocontrollers überprüfen. Das Testframework führt die Tests nacheinander aus und setzt nach jedem abgeschlossenen Test die Ausführung des Mikrocontrollers zurück. Auf diese Art können die Studierenden von extern die Mikrocontroller programmieren und zusätzlich durch definierte Testszenarien die eigenen Programme testen, ohne vor Ort sein zu müssen [9].

5 Fazit

Zusammenfassend lässt sich das langfristige Projekt cITicar als ein sehr vielseitig einsetzbares und gut angenommenes Lehrprojekt beschreiben. Bei der Durchführung der Lehrmodule fällt jedoch sowohl in der Präsenz- wie auch in der Onlinelehre immer wieder auf, dass an einigen Stellen das Grundlagenwissen zum Bearbeiten bestimmter Aufgaben am Fahrzeug noch nicht weitgehend genug vorhanden ist, sodass ein Zeitkonflikt zwischen Grundlagenlehre und Projektarbeit am Fahrzeug entsteht. Auch wenn die Containerisierung der Arbeitsumgebung den Zeitaufwand bei der Einrichtung des cITicar-Systems deutlich reduziert hat, benötigen die Studierenden besonders in der Anfangsphase im Projekt sehr viel Unterstützung, um eigenständig mit den Projektarbeiten beginnen zu können. Die Einrichtung und Verwaltung der verschiedenen Werkzeuge zur Unterstützung der Onlinelehre hat ebenfalls viel Zeit auf Seiten der Mitarbeitenden in Anspruch genommen. Abschließend lässt sich feststellen, dass die teilnehmenden Studierenden über viele Semester ein starkes Teamgefüge entwickelt haben und eine Weiterführung des Projekts am Fachbereich fest geplant ist.

6 Ausblick

Das Projekt cITicar bietet immer wieder neue Möglichkeiten, um nach und nach weitere Module oder sogar Fachbereiche im Projekt zu integrieren. Auch technologisch entwickelt sich das Grundfahrzeug immer weiter, woran sich die verschiedenen Projektarbeiten und Lehrveranstaltungen orientieren können. Ein wechselseitiger Einfluss vom Projekt auf die Lehre und umgekehrt sorgt kontinuierlich für die Verbesserung von beiden.

Verstärkter Fokus auf Machine Learning

Die aktuelle Bildverarbeitung des cITicars basiert komplett auf klassischer Bildverarbeitung [10]. Typische Operationen, wie das Anwenden von Filtern oder der Einsatz von Hough-Transformation und RANSAC-Algorithmen könnten durch alternative Ansätze aus dem Bereich des Machine Learnings ersetzt werden. Erste Schritte in modulbegleitenden Projektarbeiten in die Richtung von Neuronalen Netzen wurden bereits getätigt und sollen in der Zukunft weiter ausgebaut werden.

Vorgeschaltete Hardwareemulation mittels Renode

Wie in Abschnitt 4 beschrieben, können die Studierenden ihren Programmcode direkt von Zuhause auf die Mikrocontroller des cITicars übertragen, um Debuggen und Testen zu können. Hier kann eine weitere Zwischenebene eingeführt werden, welche zunächst den neuen Programmcode in einer vorgeschalteten Hardwareemulation auf Basis des Renode¹⁰-Frameworks testet. Teile des Fahrzeugs wurden bereits für die Emulation modelliert und konfiguriert [11], sodass im weiteren Projektverlauf ein zweistufiges Testen, erst in der Emulation und anschließend auf dem echten Fahrzeug, realisiert werden kann. Dies minimiert das Risiko von Hardwarebeschädigungen und nimmt den Studierenden die Hemmschwelle, ihren Code auszuprobieren.

¹⁰ <https://renode.io/>

Erweiterung des Testaufbaus um weitere Fahrzeuge

Um eine höhere Flexibilität im Umgang mit Testanfragen der Studierenden zu erreichen, soll das Testsystem um weitere Versuchsaufbauten des cITicar erweitert werden. Hierbei könnten Teilaufbauten für bestimmte Hardwaregruppen, ebenso wie vollständige Komplettsysteme des Fahrzeugs realisiert werden. Wird beispielsweise für einen spezifischen Test nur die Hardware rund um einen speziellen Sensor oder einen Motor benötigt, kann sich direkt auf den Minimalaufbau verbunden werden, ohne unnötig weitere Ressourcen zu blockieren. Sollte ein Fahrzeug oder eine Testeinheit bereits von den Studierenden verwendet werden, können weitere Anfragen zum Testen und Debuggen direkt über das Testframework an eine freie Teststation weitergeleitet werden. Je nach Bedarf kann das System immer weiter skaliert werden, um allen Anforderungen für die Testaufgaben zu genügen und Wartezeiten bei der Entwicklung zu dezimieren.

Integration weiterer Studiengänge in das Projekt

Als weiterer Schritt im Lehrprojekt cITicar kann die Einbindung des fachbereichseigenen Studiengangs *Social Media Systems (SMS)* realisiert werden. Die Studierenden können das cITicar mithilfe von geplanten Kampagnen und Auftritten in Social Media in der Außendarstellung verstärken und so weitere Studierende für das Projekt begeistern. Hieraus könnte resultieren, dass weitere Fachbereiche der THM Interesse am Projekt cITicar zeigen und das Projekt ebenfalls in die eigene Lehre mit einbetten. Zusätzlich können Studierende mit Hinblick auf die Finanzierung des Projekts auf Sponsoring-Suche gehen. Solch ein Projekt kann ebenfalls im Studiengang SMS durch die dem Studiengang inhärente wirtschaftliche Ausrichtung oder auch durch Studierende des Fachbereichs Wirtschaft angegangen werden. Weiterhin denkbar wären beispielsweise eine Partizipation am Projekt von Studiengängen wie Maschinenbau, Mechatronik (Fachbereiche M und ME) oder Elektro- und Informationstechnik (Fachbereiche EI und IEM).

7 Literatur

- [1] Deloitte, „Urbane Mobilität und autonomes Fahren im Jahr 2035 | Welche Veränderungen durch Robotaxis auf Automobilhersteller, Städte und Politik zurollen“, 2019. Zugriffen: Aug. 01, 2020.
- [2] R. Berger, „Anzahl der Patente mit Bezug zum autonomen Fahren weltweit von 2015 bis 2020“, 2020. Zugriffen: Okt. 01, 2021.
- [3] IPlytics GmbH, „Anzahl der angemeldeten Patente im Bereich autonomes Fahren weltweit in den Jahren 2009 bis 2019*“, Statista GmbH, Okt. 16, 2019. Zugriffen: Aug. 15, 2020.
- [4] ZSW, „Weltweite Bestandsentwicklung von Elektroautos in den Jahren 2012 bis 2019“, Statista GmbH, 2020. Zugriffen: Aug. 15, 2020.
- [5] M. Nolte, R. Graubohm, T. Form, und M. Maurer, „Carolo-Cup — Zehn Jahre automatisiertes Fahren im Maßstab 1:10“, Sonderprojekte ATZ/MTZ, Bd. 22, Nr. S2, S. 16–21, Dez. 2017.
- [6] M. Nolte, T. Form, S. Ernst, R. Graubohm, und M. Maurer, „The Carolo-Cup Student Competition: Involving Students with Automated Driving“, in 2018 12th European Workshop on Microelectronics Education (EWME), Braunschweig, Germany, Sep. 2018, S. 95–99.
- [7] J. Czekansky, „Konzeption und Realisierung der Hardware und Software eines autonomen Modellfahrzeugs“, Masterthesis (unveröffentlicht). Institut für Technik und Informatik, Technische Hochschule Mittelhessen, Aug. 20, 2020.
- [8] A. Werner und R. Kaiser, „Autonomes Fahren in der Lehre“, in Echtzeit und Sicherheit, H. Unger, Hrsg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, S. 39–48.
- [9] C. Haefke, „Ein Test-System für Continuous Integration auf Embedded Systems“, Bachelorthesis (unveröffentlicht). Institut für Technik und Informatik, Technische Hochschule Mittelhessen, Sep. 18, 2020.
- [10] M. Mengel, „Bildverarbeitung eines autonomen Modellfahrzeugs im Maßstab 1:10“, Bachelorthesis (unveröffentlicht). Institut für Technik und Informatik, Technische Hochschule Mittelhessen, Sep. 20, 2019.
- [11] S. Kunz, „Modellierung von Hardware-Komponenten zur Integration in eine Simulationsumgebung verteilter, eingebetteter Systeme“, Bachelorthesis (unveröffentlicht). Institut für Technik und Informatik, Technische Hochschule Mittelhessen, Sep. 17, 2020.