Verwendung des Verkehrsflusssimulationswerkzeugs PELOPS mit HiL-Funktionalität bei der Entwicklung von Fahrerassistenzsystemen

The traffic flow simulation program PELOPS with HiL-functionality as a tool for the development of advanced driver assistance systems

Dipl.-Ing. **F. Christen**, Aachen; Dipl.-Ing. **A. Benmimoun**, Aachen;
Dipl.-Ing. **K. Breuer**, Aachen; Dipl.-Ing. **D. Sandkühler**, Aachen;

Kurzfassung:
In den letzten Jahren ist die Fahrzeugführung durch das stetig zunehmende Verkehrsaufkommen zu einer komplexen Tätigkeit geworden. Dadurch bedingt ist gleichzeitig die Entwicklung einer Vielzahl von Fahrerassistenzsystemen zu beobachten, die den Autofahrer bei der Bewältigung der Fahraufgaben unterstützen. Sie entlasten ihn von monotonen Aufgaben und können in verschiedenen Fahrsituationen, wie beispielsweise beim Spurwechseln, Unachtsamkeiten seitens des Fahrers kompensieren.

Bei der Entwicklung solcher Systeme gewinnt der Einsatz der Simulation immer mehr an Bedeutung. Die Fahrerassistenzsystementwicklung stellt an eine Simulationsumgebung sehr spezifische Anforderungen. Da das Assistenzsystem mit seiner Umgebung interagiert, ist es zwingend erforderlich, dass die Simulationsumgebung neben einer detaillierten Abbildung des eigenen Fahrzeugs auch ein geeignetes Modell der Verkehrsumgebung zur Verfügung stellt. Dies leistet das Verkehrsflusssimulationsprogramm PELOPS.

Um den Entwicklungsprozess mit PELOPS von der Konzeptphase bis zur Umsetzung eines Regelsystems auf einem Steuergerät unterstützen zu können, wurde PELOPS um SiL(Software-in-the-loop)- und HiL(Hardware-in-the-loop)-Funktionalitäten erweitert. Zur Realisierung der SiL-Funktionalität kann PELOPS über eine konfigurierbare Schnittstelle mit Matlab/Simulink gekoppelt werden. Auch die PELOPS-Hardware-in-the-loop-(PHIL)-Simulation zeichnet sich durch ein hohes Maß an Flexibilität aus. Als Schnittstellen stellt PHIL eine Netzwerkanbindung, eine serielle Schnittstelle, über die beispielsweise eine Kopplung mit dSpace-Hardware vorgenommen werden kann, sowie CAN-Bus zur Verfügung.

Anhand ausgewählter Beispiele stellt der vorliegende Beitrag dar, wie flexibel die HiL-Simulation mit PELPOS an eine Entwicklungsaufgabe angepasst werden kann und welche Ergebnisse erzielt werden können.

Abstract:
During the last years a rapid increase of traffic volume in association with limited road space could be observed. This led to a raising load of the driver while achieving his driving tasks. Therefore so called Advanced Driver Assistance Systems (ADAS) are under development in recent years with an upward tendency. Such systems will relieve the driver and improve safety on the road in the future.

A rising number of computed-aided-simulation becomes important for the development and evaluation of such systems. Developing ADAS makes a specific demand on the simulation environment. ADAS interact strongly with their traffic environment. Therefore the simulation has to cover both, a detailed model of the ego vehicle and an adequate representation of the traffic environment. The traffic flow simulation program PELOPS meets these requirements.

The newly implemented extension of PELOPS with techniques of Software- and Hardware-in-the-Loop-simulation (SiL- and HiL-simulation) allows system developers a fast functional check and an assessment of new ADAS. External software can be linked to PELOPS via Ethernet-based network communication, which has already been realised for the widely used development tool MATLAB/Simulink. Besides the Ethernet interface PELOPS-HiL (PHIL) gives the developer two other possibilities for providing a HIL-environment: a serial interface which can be used for a connection to a dSpace development platform and a CAN data bus interface.

The following article describes by means of chosen examples how flexible the PHIL-simulation can be adapted to a development task and which results can be obtained .

1. Entwicklungswerkzeuge für Fahrerassistenzsysteme

Neben der Möglichkeit, Fahrerassistenzsysteme direkt in Fahrversuchen im realen Verkehr auf der Straße zu testen, werden heutzutage vermehrt computergestützte Simulationen eingesetzt. Der Einsatz der Computersimulation bietet eine Reihe von Vorteilen: u.a. kürzere Entwicklungszeiten, geringerer Material- und Personeneinsatz und hohe Reproduzierbarkeit der Fahrsituationen [2].

Während die klassische Simulation im Fahrzeugbau jedoch immer nur das zu prüfende Fahrzeug rechnerisch nachbildet, stellt die Assistenzsystementwicklung weitergehende Ansprüche an Simulationsumgebungen. Zukünftige Fahrerassistenzsysteme benötigen zur Durchführung ihrer Regelaufgabe eine möglichst vollständige Erfassung des Verkehrsumfeldes, um die aktuelle Verkehrssituation genügend genau identifizieren zu können. Beispielsweise regeln existierende ACC (Adaptive Cruise Control) Systeme i. A. nur auf ein relevantes Fahrzeug in ihrer Fahrspur, wohingegen ACC-Stop&Go-Systeme aufgrund der geringen Abstände in Stausituationen beispielsweise auch Fahrzeuge auf den Nachbarspuren erfassen müssen, um so ggf. frühzeitig Spurwechselvorgänge zu erkennen. Somit muss ein Simulationswerkzeug zur Entwicklung von Fahrerassistenzsystemen in der Lage sein, das Assistenzsystem über „virtuelle Umgebungssensoren“ mit allen erforderlichen Umgebungsdaten zu versorgen. Weitere Anforderungen an das Entwicklungswerkzeug sind eine realistische Abbildung von Verkehr, Echtzeitfähigkeit und die Bereitstellung geeigneter Hard- bzw. Softwareschnittstellen für den Einsatz sowohl am Arbeitsplatz als auch bei Fahrversuchen.

Mit der von ika/fka entwickelten submikroskopischen Verkehrsflusssimulation PELOPS war bereits in der Vergangenheit eine realistische Abbildung des Verkehrs möglich. Dazu bildet PELOPS in der Simulation die drei wesentlichen Elemente des Verkehrs – Strecke/Umwelt, Fahrer und Fahrzeug – mit ihren Wechselwirkungen ab. Heute steht eine neue Version von PELOPS zur Verfügung, die echtzeitfähig ist und für den Einsatz als Hardware-in-the-Loop Werkzeug modifiziert wurde. Diese Version ist in der Lage, Daten über verschiedene Soft- bzw. Hardwareschnittstellen bereitzustellen. Momentan sind dies TCP/IP, CAN-Bus und RS422 als Verbindung zu einem dSpace-System.

2. Das Simulationsprogramm PELOPS

Das submikroskopische, fahrzeugorientierte Verkehrsflusssimulationsprogramm PELOPS (Programm zur Entwicklung längsdynamischer, mikroskopischer Verkehrsprozesse in systemrelevanter Umgebung) wurde bei ika/fka in Zusammenarbeit mit der BMW AG entwickelt [8] [3]. Es wird heute von der fka vertrieben und gepflegt.

Das Konzept von PELOPS besteht in der Verknüpfung detaillierter submikroskopischer Fahrzeugmodelle mit mikroskopischen verkehrstechnischen Modellen, die sowohl eine Untersuchung des längsdynamischen Fahrzeugverhaltens als auch eine Analyse des Verkehrsablaufs ermöglichen. Der Vorteil dieser Methode liegt darin, alle Wechselwirkungen

zwischen Fahrer, Fahrzeug und Verkehr (vgl. Bild 1) berücksichtigen zu können. In einer modularen Programmstruktur werden die genannten Elemente modelliert und durch Schnittstellen abgegrenzt.

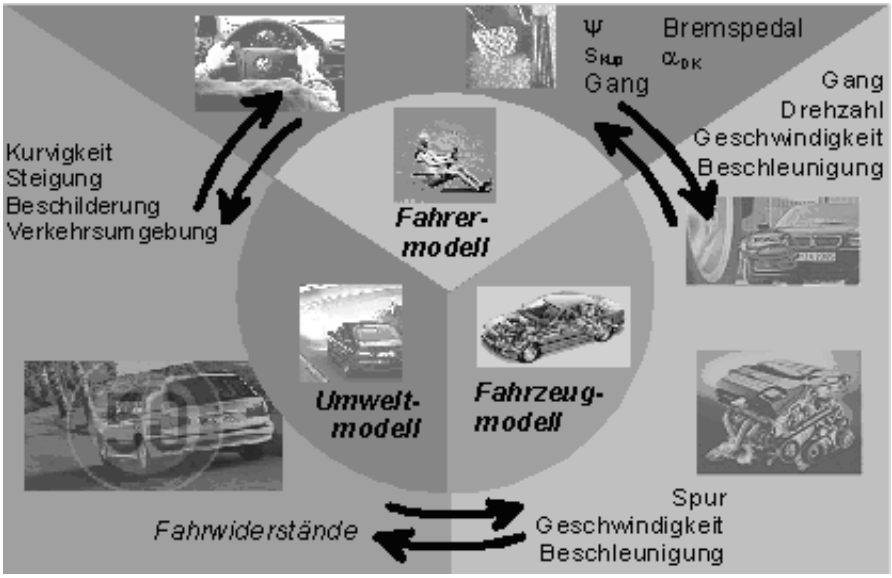

Bild 1: Das Simulationsprogramm PELOPS
Simulation system PELOPS

Die Benutzeroberfläche von PELOPS ist dreigeteilt: der Pre-Prozess ermöglicht das Zusammenstellen eines Verkehrsszenarios, der Solver beinhaltet den Berechnungskern und anhand verschiedener Post-Prozesse können die Ergebnisse visualisiert werden. Im Pre-Prozess kann der Anwender die Fahrer-Fahrzeug-Einheiten entweder einzeln aus Daten zusammensetzen, oder auf in einer Bibliothek enthaltene Standarddatensätze einzelner Komponenten bzw. ganzer Einheiten zurückgreifen. Dasselbe gilt für den Steigungsverlauf, die Kurvigkeit und die Beschilderung einer Strecke. Um bei größeren Verkehrsszenarien die Eingabe der Fahrer-Fahrzeug-Einheiten (FFE) zu vereinfachen, wurde ein FFE-Generator entwickelt, der anhand weniger Vorgaben (wie Anzahl zu erzeugender FFE, Lkw-Anteil, mittlere Geschwindigkeit, etc.) die Einzeldaten nach statistischen Randbedingungen verteilt. Mit den verschiedenen Arten des Post-Prozessing besteht nach der Berechnung die Möglichkeit, die Ergebnisse statistisch, graphisch oder als Animation aufzuarbeiten. Mit der statistischen Verarbeitung können aus den Ergebnisdaten für einzelne Messquerschnitte Fundamentaldiagramme oder auch Häufigkeitsverteilungen der Geschwindigkeiten und Abstände erstellt werden. Dadurch besteht eine einfache Möglichkeit des Ergebnisvergleiches mit realen Messschleifen im Straßenverkehr. Die graphische Aufbereitung ermöglicht für die einzelnen Fahrer-Fahrzeug-Einheiten beliebige Punkt- oder Liniendarstellungen der Ergebnisdaten. Als Animationen steht mit dem ‘Bird's Eye’ eine Betrachtung von oben auf den

Verkehrsablauf zur Verfügung. Mit den Erweiterungen aus [2] ist es nun möglich, bereits während der Simulation eine dreidimensionale Animation aus der Fahrersicht eines vor Simulationsstart ausgewählten Referenzfahrzeugs zu betrachten.

2.1. Fahrer, Fahrzeug- und Umweltmodell

Das Umweltmodell erlaubt bei Bedarf eine detaillierte Beschreibung der Einflüsse einer stationären Verkehrsumgebung. Sowohl der Verlauf der Straße in horizontaler und vertikaler Richtung über Radien und Klothoiden bzw. Steigungen, Gefälle und vertikale Ausrundungen, als auch die Anzahl und die Breite der Spuren wird angegeben. Zusätzlich zu diesen geometrischen Daten können Verkehrszeichen, Spurmarkierungen sowie Umweltbedingungen über Parameter wie etwa Nässe, Glätte und Sichtweite vorgegeben werden. Die aktuellen Verkehrsbedingungen für ein Fahrzeug ergeben sich dann aus der Anzahl der umgebenden Fahrzeuge sowie deren Abständen und Geschwindigkeiten. Um bestimmte Verhaltensweisen im Verkehr zu provozieren oder vorgegebene Fahrzyklen nachzufahren, können einzelnen Fahrer-Fahrzeug-Einheiten auch bestimmte Bewegungsprofile vorgeben werden. Das Einsetzen der Fahrzeuge am Streckenbeginn kann sowohl anhand von makroskopischen (Verkehrsstärke, Durchschnittsgeschwindigkeit und prozentualer Lkw-Anteil je Spur) als auch von mikroskopischen (Fahrzeugart, Einsetzzeitpunkt, Geschwindigkeit und Spur je Fahrzeug) Verkehrsdaten erfolgen. Des weiteren besteht auch die Möglichkeit, die Simulation als "endloses Band" durchzuführen, bei dem das Streckenende mit dem Streckenanfang verbunden ist.

Das Fahrzeugmodell basiert auf dem ‘Ursache-Wirkungs-Prinzip’, bei dem eine Berechnung der Antriebskraft ausgehend vom Motorbetriebspunkt über Kupplung, Getriebe und Differential zu den Rädern erfolgt, wo die Antriebskraft dann mit den Fahrwiderständen bilanziert wird. Der Betriebspunkt wird über die Änderung des Motormomentes (Ursache) gewechselt. Aus der dadurch verursachten Beschleunigung und Geschwindigkeitsänderung resultiert unter Berücksichtigung der Elemente des Antriebsstranges die Motordrehzahl (Wirkung). Als Getriebearten sind das konventionelle Handschalt- sowie Wandlerautomatikgetriebe implementiert. Für Nutzfahrzeuge können zusätzlich einer Motorbremse oder ein Retarder im Antriebsstrang abgebildet werden. Diese detaillierte Abbildung des Fahrzeugs unter Verwendung des Ursache-Wirkungs-Prinzips lässt die Untersuchung regelungstechnischer Einrichtungen wie beispielsweise eines ACC zu. Zusätzlich zu der beschriebenen Darstellungsform von Fahrzeugen, die in PELOPS als ‘Realfahrzeuge’ bezeichnet werden, existiert auch die Möglichkeit, mit einem weniger komplexen Fahrzeugmodell zu arbeiten, um bei

geringerem Rechenbedarf Verkehrsphänomene zu untersuchen. Solchermaßen abgebildete Fahrzeuge werden als ‘synthetische’ Fahrzeuge bezeichnet. Sie lassen sich über einen reduzierten Datensatz beschreiben, der sich aus den Parametern Fahrzeugtyp, maximale Motorleistung, maximale Verzögerung, Gesamtgewicht, Luftwiderstand und Radwiderstand zusammensetzt [10]. Im Fahrzeugmodell wird mit der für jedes synthetische Fahrzeug vorgegebenen maximalen Motorleistung zunächst die Höchstgeschwindigkeit berechnet. Aus dem Verhältnis zwischen der aktuellen und der maximalen Fahrzeuggeschwindigkeit wird mit Hilfe eines statistischen Ansatzes [3] die aktuelle Übersetzung und der Massenfaktor bestimmt. Damit berechnet sich die maximal mögliche Beschleunigung im aktuellen Fahrzustand unter der vereinfachten Annahme einer linearen Motorkennung mit Hilfe der Fahrwiderstandsgleichungen bei aktueller Geschwindigkeit und bei Höchstgeschwindigkeit [2].

In [2] ist das PELOPS Fahrzeugmodell dahingehend erweitert worden, dass nun auch das querdynamische Fahrzeugverhalten abgebildet wird. Die Unterscheidung zwischen synthetischen und Realfahrzeugen wird auch bei der Berechnung der Querdynamik weiter aufrecht erhalten. Synthetische Fahrzeuge werden anhand eines vereinfachten Einspurmodells für stationäre Kreisfahrt abgebildet, für Realfahrzeuge wird das Einspurmodell für instationäres Verhalten verwendet.

Fahrzeuge können mit einem Sensor zur Erfassung des Fahrzeugumfelds ausgestattet werden, welches dem Anwender erlaubt, beliebig viele Sensoren an beliebigen Stellen des Fahrzeugs zu positionieren. Die Sensoreigenschaften, wie beispielsweise Taktrate, Öffnungswinkel, Reichweite und Fehlerrate können frei vorgegeben werden.

Die Verbindung zwischen der Fahrzeug- und der Verkehrssimulation stellt das Fahrermodell dar. Es ist in ein Verhaltens- und ein Handlungsmodell gegliedert. Das Verhaltensmodell besteht wiederum aus zwei Teilen, dem Folge- und dem Spurwechselmodell. Das Folgemodell, welches ursprünglich auf Arbeiten von Wiedemann [11] basiert, beschreibt den Verkehr auf einer einspurigen Richtungsfahrbahn, auf der es keine Möglichkeit zum Überholen und Spurwechseln gibt. Im Vergleich zum Wiedemann-Modell ist es jedoch stark erweitert worden. So ist es etwa in der Lage, Fahrerreaktionszeiten, Ampelannäherungen, kooperatives Verhalten und Reaktionen auf andere als das direkt vorausfahrende Fahrzeug abzubilden. Das Spurwechselmodell deckt alle Verkehrssituationen ab, die auf mehrspurigen Straßen und im innerstädtischen Verkehr auftreten. Es beinhaltet nicht nur die klassischen Spurwechselsituationen, wie das Überholen auf mehrspurigen Richtungsfahrbahnen, das Aus-

weichen vor Hindernissen und das Spurwechseln zum Verfolgen einer Route durch ein Straßennetz, sondern auch taktische Überlegungen wie z. B. das Blinken, um ein Hereinlassen in enge Lücken zu provozieren [7]. Im Verhaltensmodell werden die Parameter der lokalen Fahrstrategie aus dem aktuellen Fahrzustand und der Fahrzeugumgebung bestimmt. Die Parameter der lokalen Fahrstrategie sind eine vom Fahrer gewünschte Beschleunigung, die gewählte Fahrspur und ggf. der einzulegende Fahrgang. Im Handlungsmodell schließlich werden diese Parameter in fahrzeugseitige Stellgrößen wie Lenkbewegung, Pedalbetätigung, Gangwahl und Setzen des Blinkers umgesetzt [9]. Fahrer- und Umweltmodell ermöglichen zusammen die Generierung von virtuellem Verkehr.

2.2. In-the-Loop Funktionalitäten

Mit den Erweiterungen aus [2] ist es möglich, PELOPS als Rapid-Prototyping-Werkzeug in der Assistenzsystementwicklung einzusetzen. Somit ist es bei der Simulation neuer Systemkonzepte nicht mehr zwingend erforderlich, Algorithmen und Funktionen direkt als Quellcode in PELOPS einzubinden. Sowohl Hardwarekomponenten als auch Software können mit PELOPS in einer gekoppelten Simulation betrieben werden. Dazu stellt PELOPS als Schnittstellen Ethernet-Netzwerkanbindungen, serielle RS422-Schnittstellen, über die beispielsweise eine Kopplung mit dSpace-Hardware vorgenommen werden kann, sowie CAN-Bus zur Verfügung.

Die Durchführung von Hardware-in-the-Loop(HiL)-Simulationen verlangt generell nach einer Simulation in Echtzeit. Dies bedeutet, dass eine Zeiteinheit in der Simulation gleich lange andauert wie in der Realität. Dies ist notwendig, da die zu testenden Komponenten den gleichen Randbedingungen ausgesetzt werden müssen wie bei deren Einsatz in einem vollständig realen Umfeld. So zeigt sich beispielsweise, ob die Arbeitsgeschwindigkeit eines geprüften Steuergerätes für seine Aufgabenstellung ausreicht, die verwendete Datenübertragungsrate zwischen einzelnen Komponenten eines Systems hoch genug ist, oder mechanische Stellglieder in ausreichend kurzer Zeit präzise genug angesteuert werden können [2].

Im Hardware-in-the-Loop Fall erfolgt die Synchronisierung von PELOPS mit einer realzeitlichen Bezugsgröße. Diese Bezugsgröße kann entweder durch den in jedem PC enthaltenen quarzgesteuerten Echtzeituhr-Baustein oder von einer an der Simulation beteiligten Komponente, die entsprechend ausgestattet ist, zur Verfügung gestellt werden. Bei einem externen Zeitbezug muss jedoch sichergestellt sein, dass der entsprechende Lieferant in der Lage ist,

seine Zeitbasis zu kontrollieren und etwaige Fehler selbständig zu detektieren, da sonst die gesamte Simulation falsch ablaufen und die Ergebnisse verfälschen würde [2].

Steht keine Komponente mit externer Echtzeitbasis zur Verfügung, verwendet die HiL-Variante von PELOPS den Uhrbaustein des PCs. Dieser Baustein ist autonom und daher unabhängig von der Rechenbelastung des Prozessors, so dass von einer konstanten Zeitbasis ausgegangen wird. Die Verwendung des Uhrbausteins als interne Zeitbasis wird unter anderem bei einer HiL-Simulation von PELOPS mit einem direkten Anschluss an den CAN-Bus notwendig. Da dieser Datenbus keine festen Zeitschlitze für jeden Busteilnehmer bietet, kann er nicht als verlässliche Quelle für eine Zeitbasis angenommen werden [2].

Um fehlerhafte Ergebnisse zu vermeiden, wird die Simulation bei Verlassen der Echtzeit generell sofort abgebrochen und der Anwender über diesen Umstand informiert. Da dieser Fall in der Regel nur durch zu hohe Rechenbelastung eintritt, kann der Benutzer dann entweder sein Simulationsszenario etwas vereinfachen, oder die Rechenleistung der entsprechenden Komponente erhöhen. Bei einem PC für PELOPS kann dies beispielsweise durch das Stoppen gerade nicht notwendiger Prozesse erreicht werden [2].

Im Gegensatz zur HiL-Simulation muss eine Software-in-the-Loop(SiL)-Simulation in der Regel nicht in Echtzeit ablaufen, da die Zeitbasis der mit Daten zu versorgenden Software ebenfalls nicht in Echtzeit abläuft. Um die Konsistenz der Simulationsdaten zu gewährleisten, müssen in diesem Fall aber alle beteiligten Programme synchronisiert werden. Die notwendige Synchronisierung aller Teilnehmer wird an den Schnittstellen für den Datenaustausch vorgenommen, indem sowohl PELOPS als auch die angekoppelte Software immer beliebig lange auf die Daten des aktuellen Zeitschrittes der jeweils anderen Komponente warten. Zur Realisierung einer solchen Synchronisierung ist auf beiden Seiten eine einheitliche Zeitschrittweite bei der Berechnung bzw. an der Datenschnittstelle notwendig [2].

Eine Schnittstelle zu MATLAB/Simulink ist in PELOPS bereits vorhanden. Der Datenaustausch während der SiL-Simulation geschieht über eine ethernetbasierte Netzwerkkommunikation. Prinzipiell gibt es zwei Möglichkeiten, solche Verbindungen herzustellen. Einerseits ist dies die Punkt-zu-Punkt-Variante, bei der sowohl dem sendenden Computer als auch der empfangenden Einheit die IP-Adresse / Port-Nummer des jeweils anderen bekannt sein muss. Diese Variante verschickt die Datenpakete immer nur an die dediziert angegebenen IP-Adressen, die auch außerhalb eines Subnetzes liegen können, sofern eine Firewall dies

nicht verhindert [1]. Der Datenaustausch erfolgt bei einer Punkt-zu-Punkt-Verbindung in der Regel mit dem Transmission Control Protocol (TCP). Es soll sicherstellen, dass die Datenblöcke (Segmente) unversehrt und in der richtigen Reihenfolge beim Empfänger ankommen.

Eine zweite Verbindungsvariante realisiert den Datenaustausch durch die Verwendung sogenannter Multicast-Pakete. Dieses auch als „Broadcast“ bezeichnete Verfahren sendet die Pakete nicht an explizit bekannte IP-Adressen, sondern versieht die Nachrichten mit einer IP-Nummer aus dem Multicast-Adressraum, der sich von 224.0.0.0 ... 239.255.255.255 erstreckt. Dadurch ist es im Gegensatz zur ersten Variante auch möglich, Punkt-zu-Multipunkt-Verbindungen herzustellen, so dass z.B. ein PELOPS-Prozess mit mehreren MATLAB-Prozessen (auf einem oder mehreren Rechnern) kommunizieren kann, ohne dass ihm dafür die exakte Netzwerkkonfiguration der Gegenseite bekannt sein muss [1]. Im Gegensatz zur Punkt-zu-Punkt-Verbindung erfolgt der Datenaustausch bei einer Punkt-zu-Multipunkt-Verbindung nicht mittels TCP sondern anhand des User Datagram Protocol (UDP). UDP ist eine funktionsreduzierte Alternative zu TCP und kontrolliert die Datenübertragung nur minimal. Weder stellt es sicher, dass Datenblöcke tatsächlich beim Empfänger ankommen, noch kümmert es sich darum, ob sie in der richtigen Reihenfolge eintreffen. Dies ermöglicht jedoch eine höhere Datenrate als bei TCP, was den Anwendungen zugute kommt.

Beide soeben beschriebenen Verbindungen können sowohl zwischen räumlich getrennten Computern als auch innerhalb eines einzigen Computers zwischen verschiedenen Prozessen erfolgen. Während der Entwicklung und Nutzung hat sich die zweite Variante als die unproblematischere und leistungsfähigere herausgestellt, da keine Übertragungsbandbreite eines Netzwerks die Ausführungsgeschwindigkeit einer gekoppelte Simulation künstlich verringert.

3. PELOPS als HiL-Werkzeug – Anwendungsbeispiele

Nachdem in Abschnitt 2 das Simulationswerkzeug PELOPS und insbesondere seine In-the-Loop-Funktionalität detailliert erläutert worden ist, wird im Folgenden anhand einiger Anwendungsbeispiele aus aktuellen Entwicklungsprojekten der fka die Vielseitigkeit der PELOPS-HiL-Simulation dargestellt. Die Beispiele zeigen, wie reale Versuchsfahrzeuge in den virtuellen PELOPS-Verkehr eingebunden werden (Abschnitt 3.1) oder ACC-Steuergeräte einen virtuellen Versuchsträger automatisch längsführen (Abschnitt 3.3). Darüber hinaus wurde das PELOPS-Fahrermodell zur Steuerung eines realen Versuchsfahrzeugs verwendet (Ab-

schnitt 3.2). Abschnitt 3.4 stellt dar, wie dSpace-Hardware mit der PELOPS-Simulation gekoppelt werden kann.

3.1. Integration eines Versuchsträgers in die PELOPS-Simulation

Wie die Kapitelüberschrift schon suggeriert, wird in dieser Variante nicht ein Fahrerassistenzsystem alleine, sondern direkt in einem entsprechend ausgestatteten Versuchsträger in die PELOPS-Simulation integriert. Das folgende Bild veranschaulicht, wie sich die unterschiedlichen Simulationsmodule je nach Anforderung zwischen Simulation und Realität aufteilen können.

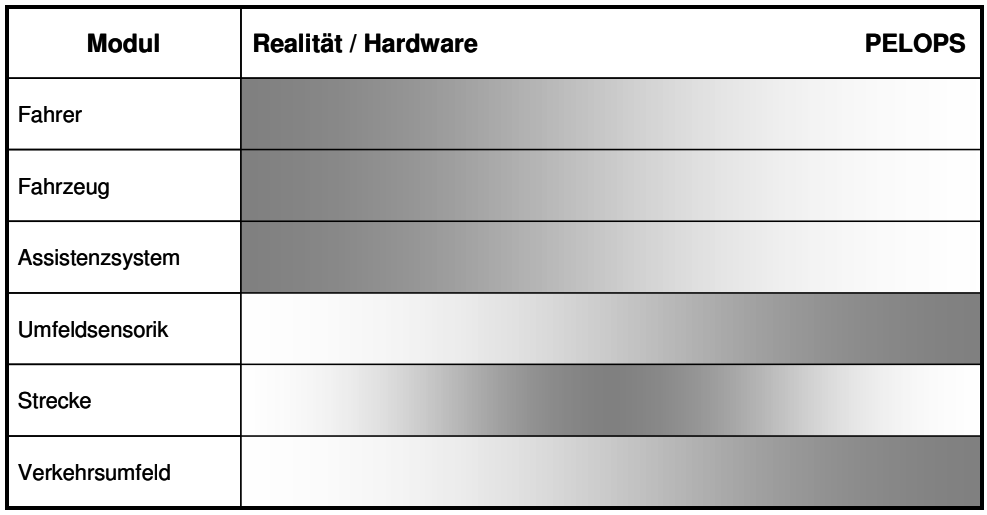

Bild 2: Aufteilungsarten der Simulationsmodule zwischen Realität/Hardware und PELOPS
Possible kinds of partitioning of the modules between PELOPS and reality/hardware

Die Verbindung zwischen Simulationsumgebung und Fahrzeug erfolgt dabei über mehrere CAN-Bus Anschlüsse, da die im Rahmen der Fahrerassistenzsystementwicklung eingesetzten Versuchsträger meist über zwei oder mehr voneinander unabhängige CAN-Bus Netzwerke verfügen. Einerseits ist dies in der Regel ein sogenannter Antriebsstrang-CAN, wie er üblicherweise in modernen Serienfahrzeugen vorzufinden ist. Gewöhnlich sind an einen solchen Bus alle den Antriebsstrang betreffenden ECUs (Electronic Control Unit) wie etwa Motorelektronik, elektronische Getriebesteuerung oder Fahrdynamikregelung angeschlossen. Auf diesem Bussystem werden sämtliche Daten übertragen, die zum (normalen) Fahrbetrieb eines Fahrzeugs notwendig sind [2].

Daneben existiert häufig ein zweiter eigenständiger Sensor-CAN, an dem die für die Entwicklung eines Fahrerassistenzsystems (FAS) notwendige Umfeldsensorik angeschlossen wird. Diese in Versuchsfahrzeugen für Fahrerassistenzsysteme häufiger vorzufindende Maßnahme (vgl. z.B. [6]) erlaubt den Entwicklern eine größere Flexibilität beim Einsatz neuer Komponenten, ohne gleichzeitig Probleme im normalen Fahrzeugbetrieb zu riskieren [2].

Um einen derart ausgerüsteten Versuchsträger in eine PELOPS-Simulationsumgebung einzubinden, muss die Hardwarebasis von PELOPS ebenfalls zwei voneinander getrennte CAN-Bus Anschlüsse bereitstellen. Dies wird mit Hilfe von zwei Infineon C167CS Mikrocontrollerboards realisiert. Diese sind in einem zusätzlichen Gehäuse untergebracht, welches insgesamt drei CAN-Bus Buchsen aufweist und für den Fahrzeugeinsatz mit einer stabilisierten DC/DC-Spannungswandlung von 9..18V auf die für Mikrocontroller notwendigen 5V ausgerüstet ist. Ein CAN ist dabei für die Verbindung zu PELOPS vorgesehen, während die beiden anderen jeweils einen CAN-Bus des Versuchsträgers bedienen. Dieser in Bild 3 skizzierte Ansatz bietet den Vorteil, dass die für den jeweiligen Anwendungsfall (hier also der FAS-Versuchsträger) notwendige Datenkonditionierung und -verteilung (hier Sensor- und Antriebsstrang-CAN) auf den Mikrocontrollern stattfindet, während PELOPS die Simulationsdaten immer in für den Simulationsablauf optimierter Weise auf einem eigenen CAN-Anschluss beiden Mikrocontrollern bereitstellt. Der PELOPS-CAN wird beispielsweise zur Senkung der durch die Datenübertragung auftretenden Latenzzeiten mit der höchstmöglichen CAN-Datenrate von 1 MBit/s betrieben, was durch die kurze Kabelverbindung zwischen den Mikrocontrollern und dem CAN-Dongle am PELOPS-Rechner kein Problem darstellt (vgl. [5]). Dagegen werden die beiden anderen CAN-Bus Anschlüsse mit im Fahrzeugeinsatz üblichen 500 kBit/s betrieben [2].

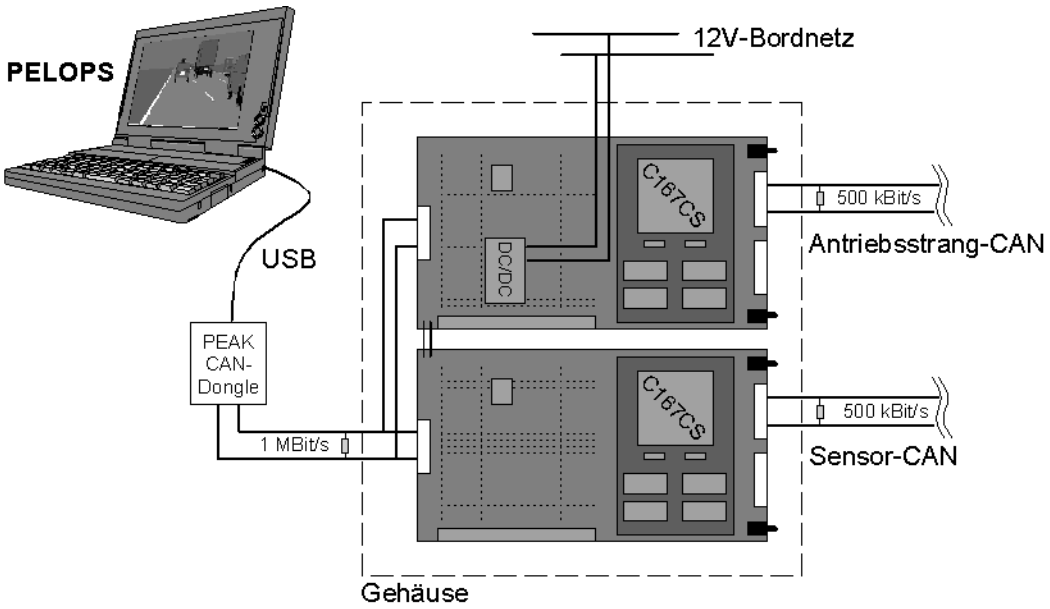

Bild 3: Prinzip der Kopplung von PELOPS mit Antriebsstrang- und Sensor-CAN in FAS Versuchsträgern [2]
Coupling principle of PELOPS with the power train- and sensor-CAN in ADAS test vehicles [2]

Die von virtuellen Abstandssensoren in der Simulation bestimmten Objektdaten (Abstände in Fahrzeuglängsrichtung und dazu orthogonal, Relativgeschwindigkeit, relative Fahrspur) werden von PELOPS auf den CAN gestellt und von einem der beiden Mikrocontroller so aufbereitet, dass sie für die im Versuchsträger eingesetzte FAS-ECU scheinbar von einem realen Radarsensor stammen. Zur exakten Abbildung des Sensorverhaltens bei der Datenbereitstellung passt die auf dem C167 laufende Software die PELOPS-Simulationsdaten in Auflösung und Bereitstellungsrate an einen realen Sensor an, und versendet die entsprechenden CAN-Nachrichten auf dem Sensor-CAN. Durch diese Aufgabenverteilung zwischen PC und Mikrocontroller wird die Verkehrssimulation nicht durch den zusätzlichen Rechen- und Synchronisationsaufwand weiter belastet, was sich für die Einhaltung der Echtzeit positiv auswirkt [2].

Die Bewegungsdaten des Versuchsfahrzeugs werden durch die bordeigenen Sensoren erfasst und stehen auf dem Antriebsstrang-CAN zur Verfügung. Der C167-Mikrocontroller erfasst die Signale der Gierrate, Quer- und Längsbeschleunigung sowie der Fahrzeuggeschwindigkeit und stellt diese Größen asynchron im für PELOPS geeigneten Takt, der in der Regel bei 100Hz liegt, auf dem 1Mbit/s CAN bereit [2].

Mit den vom Mikrocontroller bereitgestellten Bewegungsdaten des Versuchsträgers kann die Verkehrssimulation die neue Fahrzeugposition für den nächsten Zeitschritt berechnen. Während Gierrate, Längsbeschleunigung und Geschwindigkeit in Richtung der Fahrzeuglängsachse direkt verwendbar sind, können die zur Bestimmung der Bahnkurve notwendige Bahnwinkelrate und der Schwimmwinkel nur durch Berechnung gewonnen werden [2].

Bild 4: Konzept der HiL-Kopplung
Concept of the HiL-coupling

Bild 4 verdeutlicht das hier erläuterte Konzept der HiL-Kopplung. Ein Fahrer des FAS-Versuchsträgers erhält neben seinen direkten Sinneswahrnehmungen beispielsweise von Ruck oder Beschleunigung über die in Echtzeit mitlaufende Animation der Verkehrsumgebung aus Fahrersicht eine Rückmeldung über das Verhalten der von ihm entwickelten FAS-Algorithmen [2].

3.2. Das Fahrermodell als Fahrerassistenzsystem

Bei der Entwicklung von sicherheitsrelevanten Assistenzsystemen gibt es mehrere bisher ungelöste Probleme. Das Hauptproblem ist die zuverlässige Erkennung von potentiell gefährlichen Situationen, so dass das System eine Warnung an den Fahrer ausgeben kann bzw. auf eine für den Fahrer verständliche Art und Weise intervenieren kann. Für die erste Generation von Assistenzsystemen war dies wichtig relevant, da es sich hierbei hauptsächlich um Komfortsysteme handelt, während der Sicherheitsaspekt erst jetzt immer mehr in den Vordergrund rückt. So wird beispielsweise beim ACC die Kaskadenregelung als Regelungsstruktur bevorzugt. Weil hierbei eine fixe Parametrisierung nicht allen Verkehrssituationen gerecht werden kann, müssen die Regelungsparameter situationsgerecht angepasst werden. Aufgrund der geringen Informationsverfügbarkeit ist die Situationsklassifizierung nicht in jedem Fall ausreichend. Das daraus resultierende Fahrerverhalten ist deshalb für Fahrer oft nicht verständlich. Für das ACC ist dies zunächst nicht ausschlaggebend, aber für sicherheitsrelevante Systeme müssten komplexe Situationsanalysen durchgeführt und entsprechende Regelungsalgorithmen entwickelt werden.

Neben der Entwicklung solcher Algorithmen besteht die Möglichkeit, auf Fahrermodelle zurückzugreifen und diese für die Situationsklassifizierung und Bewertung zu nutzen. Im folgenden wird ein Ansatz beschrieben, bei dem das PELOPS-Fahrermodell die längsdynamischen Regelung eines realen Versuchsträgers übernimmt.

In Bild 5 ist dargestellt, wie im ersten Schritt die Implementierung des Fahrermodells im Fahrzeug erfolgt. Hierbei liefern Sensoren am Fahrzeug Informationen über die Fahrzeugumgebung und den Fahrzustand. Diese werden mittels der im vorangegangenen Kapitel beschriebenen HIL-Schnittstelle an das PELOPS Fahrermodell transferiert. In PELOPS wird eine virtuelle Welt (Streckenführung, Umgebungsfahrzeuge etc.) entsprechend den erhaltenen Informationen aufgebaut, in der der virtuelle Fahrer fährt. Die Reaktion des virtuellen Fahrers (Sollbeschleunigung und Solllenkwinkel) wird über die HIL-Schnittstelle zurück ins

Fahrzeug übertragen und über einen Regler in einer Gas-/Bremspedalstellung und einer Lenkradstellung umgewandelt.

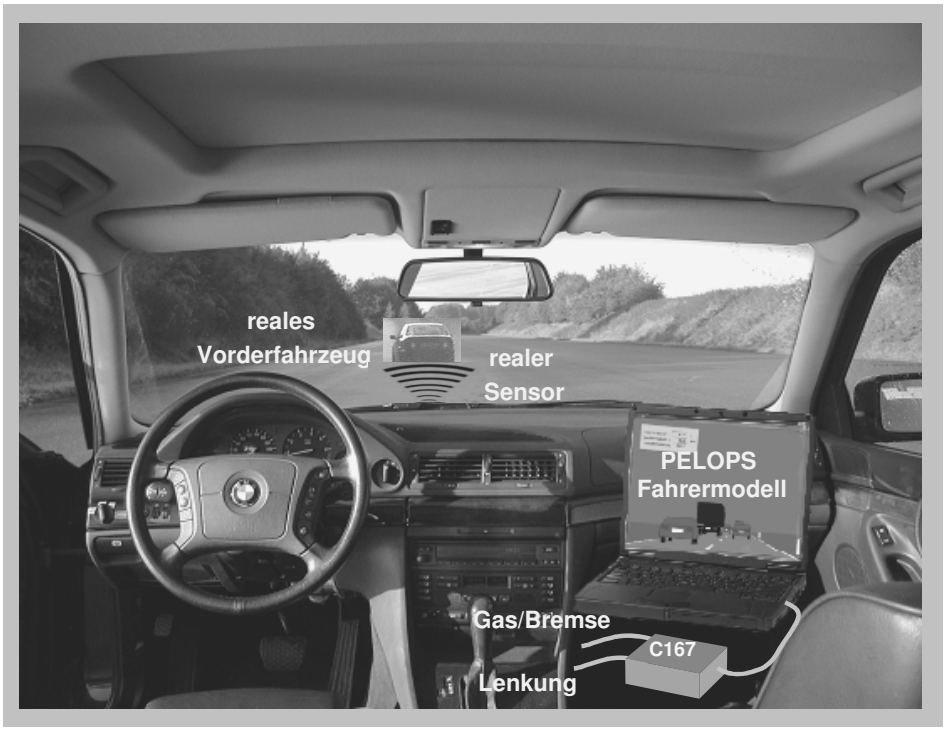

Bild 5: Konzept der Umsetzung des PELOPS-Fahrermodells als Assistenzsystem
Concept of implementation of the PELOPS-driver model as an assistance system

Das folgende Bild zeigt ein Folgemanöver, das mit dem vorgestellten Konzept geregelt wurde. Hierbei fährt zunächst das geregelte Fahrzeug in einem Abstand von etwa 75 m hinter dem Führungsfahrzeug her. Das Führungsfahrzeug beschleunigt und verzögert mehrmals bis es schließlich stehen bleibt. Zu Beginn wird die Situation “no influence” vom Fahrermodell gesetzt, da das Führungsfahrzeug schneller fährt und deshalb eine Reaktion auf dieses nicht nötig ist. In dieser Situation beschleunigt das geregelte Fahrzeug auf seine Wunschgeschwindigkeit (Relativgeschwindigkeit ist negativ, Führungsfahrzeug entfernt sich). Das Führungsfahrzeug verzögert anschließend und es wird die Situation “stop and approach” identifiziert. Dies bedeutet, dass sich das geregelte Fahrzeug annähert und das Führungsfahrzeug wahrscheinlich zum Stillstand kommt. Anschließend wird auf die Situation „approach slow jam“ geschaltet. Diese Situation beschreibt langsam kriechenden Verkehr voraus. Das Führungsfahrzeug beschleunigt wieder und es wird für kurze Zeit die Situation Folgen („following“) und dann „no influence“ detektiert. Anschließend bremst das Fahrzeug bis in den Stillstand. Hierbei regelt das Fahrermodell zunächst auf die Situation „stop and approach“ und danach auf das stehende Führungsfahrzeug („stop“), bis es ebenfalls in einem Abstand von etwa sechs Metern zum Stehen kommt. Da das Fahrermodell parametrierbar

ist, kann auch das Fahrerverhalten entsprechend sportlich oder defensiv eingestellt werden. Neben der gezeigten längsdynamischen Regelung ist mit diesem Ansatz auch eine laterale Regelung realisiert worden (Spurhaltefunktion).

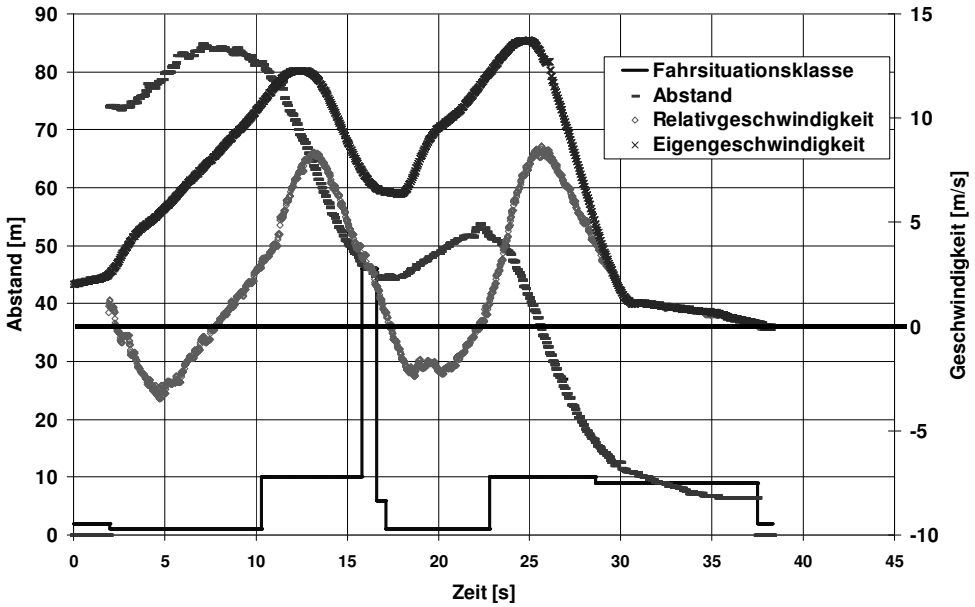

Bild 6: Ein vom Fahrermodell geregeltes Folgemanöver
A following manoeuvre, driven by the driver model

3.3. Reglerentwicklung

Nachdem in den bisher vorgestellten Anwendungen der Einsatz der PELOPS-Simulation im Versuchsfahrzeug, und damit im realen Fahrversuch, behandelt worden ist, wird in diesem Abschnitt eine reine Laboranwendung vorgestellt. Ein als Matlab-Modell vorliegendes Grundkonzept einer automatischen Längsführung für den Niedergeschwindigkeitsbereich wird in C-Code umgesetzt und zur Vorbereitung des Fahrversuchs auf ein Steuergerät appliziert. In der PELOPS-HiL-Simulation soll der Regleralgorithmus weiterentwickelt und verifiziert werden. Die Verwendung der realen Steuergeräte-Hardware schon in diesem frühen Stadium der Entwicklung hat den Vorteil, dass die typischen Beschränkungen durch die Verwendung der Zielhardware schon bei der Algorithmusentwicklung berücksichtigt werden können. Zu nennen sind hier insbesondere das Ganzzahlrechenwerk und der 16-bit-Prozessor. Daraus ergibt sich eine Einschränkung der Rechengenauigkeit, die bei der Algorithmusentwicklung berücksichtigt werden muss.

Zur Kommunikation zwischen Steuergerät und PELOPS-HiL-Simulation wird analog zur vorgesehenen Fahrzeuganwendung ausschließlich eine CAN-Schnittstelle verwendet. Um

die Software des Steuergeräts in möglichst identischer Form in Verbindung mit PELOPS und im Versuchsfahrzeug verwenden zu können, wird das CAN-Protokoll in der Simulation dem der realen Fahrzeugkomponenten entsprechend aufgebaut.

Das Steuergerät übernimmt die längsdynamische Regelung eines in PELOPS abgebildeten Realfahrzeugmodells. Die ebenfalls virtuelle Umfeldsensorik erfasst die PELOPS-Verkehrsumgebung und gibt Daten über das vorausfahrende Fahrzeug über die CAN-Schnittstelle an das Steuergerät weiter. Darüber hinaus erhält das Steuergerät über die CAN-Schnittstelle Daten über den Bewegungszustand des virtuellen ACC-Fahrzeugs. An PELOPS wird das vom Regler ermittelte Antriebs-Sollmoment versendet. Dieses wird von PELOPS in Stellgrößen für Motor bzw. Bremse übersetzt. Um alle für ACC-Anwendungen üblichen Sollvorgaben eines Reglers verarbeiten zu können, ist die PELOPS-Schnittstellenroutine entsprechend flexibel ausgeführt. In einer Vorgabedatei gibt der Anwender an, welche Art von Sollwert vom Regler an PELOPS übergeben wird. Damit kann dieser in der Simulation korrekt verarbeitet werden.

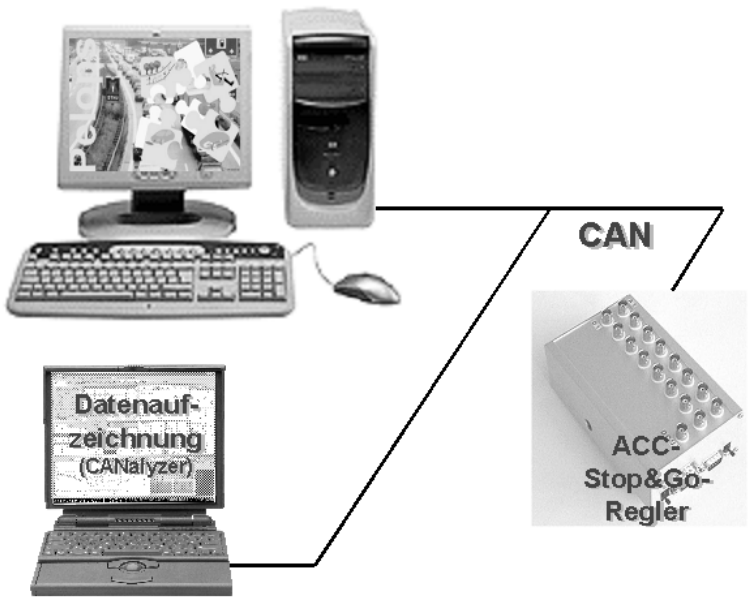

Bild 7: Struktur der Simulationsumgebung
Structure of the simulation environment

Als weitere Komponente der hier vorgestellten Simulationsumgebung wird ein Notebook als Mensch-Maschine-Schnittstelle verwendet. Dies erfolgt ebenfalls analog zum Fahrversuch. Hierüber werden Wunschgeschwindigkeit, Sollzeitlücke, Regler Ein/Aus und ggf. ein Assisted Go-Signal realisiert. Außerdem dient das Notebook zur Datenaufzeichnung. Bild 7 zeigt den Aufbau der hier vorgestellten Simulationsumgebung.

3.4. Entwicklung eines Beschleunigungsreglers

Im Rahmen von [4] wurde für das ika/fka-Versuchsfahrzeug ein integrierter Beschleunigungsregler entwickelt und implementiert. Ein solcher Beschleunigungsregler stellt die unterste Ebene eines längsdynamischen Fahrerassistenzsystems dar, indem er die von einer höheren Reglerebene (z.B. ACC) angeforderte Sollbeschleunigung mittels Motor- und Bremseneingriffen einregelt [2].

Der entwickelte Beschleunigungsregler wird auf einem Steuerrechner auf Basis eines Mikrocontrollers Infineon C167CS betrieben, welcher über zwei voneinander unabhängige CAN-Bus-Anschlüsse verfügt. Einer der beiden Anschlüsse ist im Fahrzeug für die Verbindung mit dem bereits ab Werk vorhandenen CAN-Bus vorgesehen. Der zweite Anschluss wird für den im Versuchsfahrzeug nachgerüsteten zweiten CAN-Bus (vgl. [6]) bereitgestellt, an dem unter anderem der elektronisch ansteuerbare Bremskraftverstärker betrieben wird. Weiterhin verfügt das Steuergerät über digitale und analoge Ein- und Ausgänge, über die beispielsweise das Tachosignal des Fahrzeugs erfasst oder der elektrische Verstellmotor der Drosselklappe angesteuert wird [2].

Im Sinne eines Rapid-Prototypings wurde für die Entwicklung, Fehlererkennung und -beseitigung der Hard- und Software des Beschleunigungsreglers vor dem ersten Einsatz im Fahrzeug kontinuierlich eine PELOSP-HiL-Simulationsumgebung verwendet. Da die Simulation einer Fahrzeugumgebung für den Beschleunigungsregler nicht nur CAN-Bus-Botschaften, sondern auch weitere Signalformen umfassen muss, wurde zur Signalgenerierung und -verarbeitung eine so genannten „Expansion Box“ der Firma dSpace in die Simulationsumgebung integriert. Das Echtzeitbetriebssystem der dSpace-Umgebung dient während der Simulation als Zeitreferenz für PHIL. Von der Expansion-Box verschickte Datagramme werden dazu von PHIL zusätzlich als Taktsignal genutzt. Bild 8 zeigt die Struktur dieses Versuchsaufbaus. Dabei wird das Versuchsfahrzeug vollständig in PELOPS simuliert und erhält seine Eingangsdaten für Gas- und Bremspedalstellung von einem speziell für diesen Anwendungsfall erstellten dSpace-Modell. Dieses Modell enthält die Abbildungen der Übertragungsfunktionen der realen Drosselklappenverstellmechanik und des elektronisch ansteuerbaren Bremskraftverstärkers (BKV) [2].

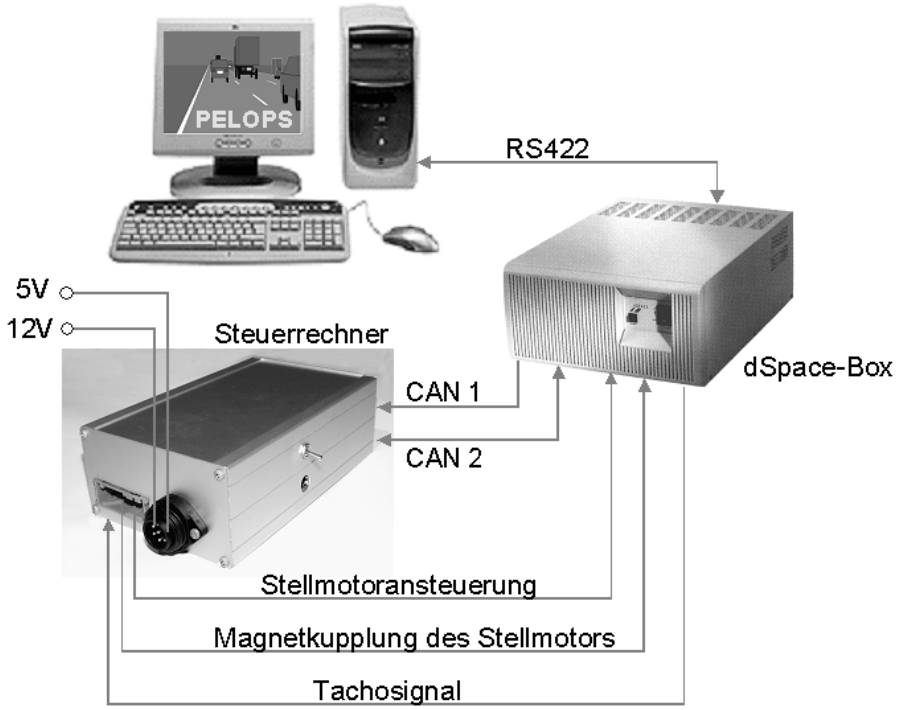

Bild 8: Struktur der Simulationsumgebung des Beschleunigungsreglers [2]
Structure of the simulation environment of the acceleration controller

Die Ansteuerung der virtuellen Drosselklappe erfolgt über zwei Analog-Digital-Wandler-Eingänge der DS2201-Schnittstellenkarte des dSpace-Systems. Dabei repräsentiert jeweils ein A/D-Eingang einen Anschluss des elektrischen Verstellmotors der realen Drosselklappe. Die Differenzspannung zwischen beiden A/D-Eingängen liefert das Eingangssignal für das Drosselklappenmodell, welches mit Hilfe eines Kennfelds abhängig von der Spannung und der aktuellen Drosselklappenposition die resultierende Verstellgeschwindigkeit ermittelt. Die Verstellgeschwindigkeit wird anschließend zur neuen Drosselklappenposition aufintegriert und via RS422-Datenverbindung an PHIL verschickt [2].

Die Regelung der Bremse erfolgt wie im realen Fahrzeug über den zweiten CAN-Bus. Der entsprechende CAN-Anschluss des Steuerrechners ist dazu mit der DS-4302 CAN-Karte der dSpace-Box verbunden. Über diese werden die vom Beschleunigungsregler verschickten CAN-Botschaften eingelesen und ausgewertet. Die Interpretation der Botschaftsinhalte entspricht dabei vollständig der CAN-Spezifikation des elektronisch ansteuerbaren Bremskraftverstärkers. Die Totzeit des realen BKV im Fahrzeug wird bei der Umrechnung des Bremsdrucksollwertes in eine PELOPS-konforme Bremspedalstellung berücksichtigt. Diese resultierende Größe wird anschließend vergleichbar zur aktuellen Drosselklappenposition über die serielle RS422-Schnittstelle an PHIL verschickt [2].

Die Bewegungs- und Betriebsgrößen des simulierten Versuchsfahrzeugs werden in PELOPS in Echtzeit errechnet und via RS422-Schnittstlle an das dSpace-Modell zurückgeschickt. Die wichtigsten Größen sind dabei die aktuelle Geschwindigkeit, Motordrehzahl und -moment, der vom Automatikgetriebe gewählte Gang und der Zustand der Wandlerüberbrückungskupplung. Die Motorbetriebsdaten werden in CAN-Botschaften für die digitale Motorelektronik umgesetzt und an das Steuergerät des Beschleunigungsreglers versendet. Die Getriebeinformationen werden analog dazu als CAN-Nachrichten der elektronische Getriebesteuerung verschickt [2].

Da im realen ika-Versuchsfahrzeug keine Geschwindigkeitsinformation auf einem Datenbus zur Verfügung gestellt wird, wertet der Beschleunigungsregler das Rechtecksignal aus, welches normalerweise von dem ab Werk verbauten Tempomaten genutzt wird. Die Simulationsumgebung stellt daher ein vergleichbar konditioniertes digitales Signal bereit. Im realen Fahrzeug liegt dieses Signal auf einem 12V-Pegel und wird von 2ms andauernden 0V-Peaks unterbrochen. Die Auftretensfrequenz dieser Spannungseinbrüche ist proportional zur Fahrzeuggeschwindigkeit [2].

4. Ausblick

Die PHIL-Simulation hat sich als sehr nützliches Werkzeug zur Entwicklung von Fahrerassistenzsystemen erwiesen. Die hier beschriebenen Anwendungsbeispiele verdeutlichen die hohe Flexibilität der PHIL-Umgebung. Neben des Einsatzes als Entwicklungs- und Prüfwerkzeug für Fahrerassistenzsysteme hat sich die PHIL-Simulation auch beim Aufbau eines Nutzfahrzeug-Fahrsimulators bewährt. Weitere Anwendungsfälle wie beispielsweise die Integration der PHIL-Umgebung in Prüfstände sind in Planung. Dabei liefert die Simulation mit geringerem Aufwand realistischere Belastungsvorgaben für die zu analysierenden Prüflinge, als dies durch die heute eingesetzten manuell erstellten Lastkollektive der Fall ist [2].

5. Literaturhinweise

[1] Breuer, K.; Christen, F.: Entwicklung von Fahrerassistenzsystemen unter Verwendung von SiL- und HiL-Techniken im Verkehrsflusssimulationsprogramm PELOPS, 11. Aachener Kolloquium Fahrzeug- und Motorentechnik 2002, Aachen 2002

[2] Breuer, K.: Verkehrsflusssimulation zur Entwicklung von Fahrerassistenzsystemen, RWTH Aachen Diss. 2004 (in Vorbereitung)

[3] Diekamp, R.: Entwicklung eines fahrzeugorientierten Verkehrsflusssimulationsprogramms, RWTH Aachen Diss. 1995

[4] Erkol, D.: Entwicklung und Implementierung eines integrierten Beschleunigungsreglers im Fahrzeug, RWTH Aachen Diplomarb. 2003

[5] Etschberger, K.: Controller-Area-Network Grundlagen, Protokolle, Bausteine, Anwendungen, 2. vollständig überarbeitete und erweiterte Auflage, Carl Hanser Verlag München Wien, München, 2000

[6] Feldhaus, J.: Ausrüstung eines Fahrzeugs mit Stromversorgung und CAN-Bus zum Aufbau eines Versuchsträger für Fahrerassistenzsysteme, RWTH Aachen Studienarb. 2002

[7] Hochstädter, A., Ehmanns, D., Breuer, K.: Das PELOPS Fahrermodell, interner technischer Bericht der Forschungsgesellschaft Kraftfahrwesen Aachen für die BMW AG, Aachen 2001

[8] Ludmann, J.: Konzept eines Verkehrssimulationsprogramms, RWTH Aachen Diplomarb. 1989

[9] Neunzig, D.: Interner Bericht zum PELOPS-Fahrermodell, Aachen 2003

[10] Weilkes, M.: Auslegung und Analyse von Fahrerassistenzsystemen mittels Simulation, RWTH Aachen Diss. 1999

[11] Wiedemann, R.: Simulation des Straßenverkehrsflusses, Schriftenreihe des Instituts für Verkehrswesen der Universität Karlsruhe, Band 8, Karlsruhe 1974