INF Informatik

Hochschule Reutlingen
Reutlingen University

Master in Human Centered Computing
„Internet of Things“
SS2016

Prof. Keller/Martinez/Steddin

- Masterprojekt -

Portable System to Detect driver drowsiness with Body Sensors.

vorgelegt von:
Paul Pasler
3. Semester

Kontaktadresse: paul.pasler@student.reutlingen-university.de
vorgelegt am: 28.06.2016

Zusammenfassung *Fahrerassistenzsysteme sind aus modernen Fahrzeugen nicht mehr wegzudenken. Müdigkeitserkennung hilft Sekundenschlaf oder müdigkeitsbedingte Unachtsamkeit zu vermeiden und verhindert somit schwere Unfälle. Systeme mit Körpersensoren zeigen in verschiedenen Studien sehr genau Ergebnisse und erkennen Müdigkeit frühzeitiger als andere Ansätze. In der vorgelegten Arbeit wurde ein solches System mit einem Elektroenzephalogramm (EEG) umgesetzt und getestet. Hierfür wurden Testdaten aufgenommen, verarbeitet und mit einem künstlichen Neuronalen Netz klassifiziert, sodass der aktuelle Status des Fahrers „Wach“ oder „Müde“ unterschieden werden kann.*

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

FAS Fahrerassistenzsystem

FASs Fahrerassistenzsysteme

ME Müdigkeitserkennung

bspw beispielsweise

RTU Reutlingen University

BS Körpersensoren

1 Einleitung

Fahrerassistenzsysteme sind innerhalb weniger Jahren von der Oberklasse, in die Mittel- und Kleinwagenklasse vorgedrungen. Die Unternehmensberatung Strategy Analytics schätzt, dass in den nächsten Jahren sechs mal soviele Fahrerassistenzsysteme verbaut werden als heute [1]. Sie bieten dem Fahrer erhöhten Komfort (Tempomat) oder die Sicherheit (Notbremsassistent). Laut der Boston Consulting Group, könnte der flächendeckende Einsatz von Fahrerassistenzsysteme, die Unfallrate um ein knappes Drittel zurückgehen lassen [2].

Zu Gruppe der Sicherheitsrelevanten Fahrerassistenzsysteme gehört auch die Müdigkeitserkennung. Beispielsweise rät die Müdigkeitserkennung „Attention Assist“ von Daimler dem Fahrer, zu gegebenen Anlass, eine Pause einzulegen und zeigt ein Kaffeesymbol im Cockpit an [3]. So wird müdigkeitsbedingte Unachtsamkeit oder Sekundenschlaf, die oftmals die Folge von schweren Unfälle sind, entgegengewirkt.

In Deutschland wurden 2015 rund 2,5 Mio. Unfälle polizeilich aufgenommen, die Zahl der Verkehrstoten liegt bei 3.450 [4]. Neben überhöhter Geschwindigkeit, zählt laut dem Deutschen Verkehrssicherheitsrat Müdigkeit zu den häufigsten Unfallursachen und ist damit für jeden fünften schweren Unfall verantwortlich [5]. Dies verdeutlicht das Potential einer frühzeitigen Erkennung von Müdigkeit und einer Meldung an den Fahrer.

Für eine sichere und korrekte Erkennung von Übermüdung ergeben sich mehrere Problemstellungen (Abb. 1.2). Anzeichen von Müdigkeit müssen vom System genau analysiert werden, um eine sichere Aussage über die Fahrtauglichkeit des Fahrers treffen zu können. Erkennt das System eine gefährliche Situation, muss der Fahrer in geeigneter Weise darauf hingewiesen werden.

Daraus ergeben sich Anforderungen für ein multimodales System zur Müdigkeitserkennung (Abb. 1.1). Es existieren bereit diverse Systeme, denen es jedoch oftmals an Komfort und Portabilität mangelt. Ziel dieser Arbeit ist die Entwicklung eines Systems zur Müdigkeitserkennung mit einem EEG-Headset. Dessen Signale werden verarbeitet und mit einem KNN klassifiziert, mit dem Ziel den Fahrer rechtzeitig vor Übermüdung und deren Folgen zu warnen. Statt dem klassischen EEGs wird ein EPOC Emotiv eingesetzt werden, dadurch soll die Beeinträchtigung des Fahrers möglichst gering gehalten werden. Auch wenn sich der Ansatz weniger für

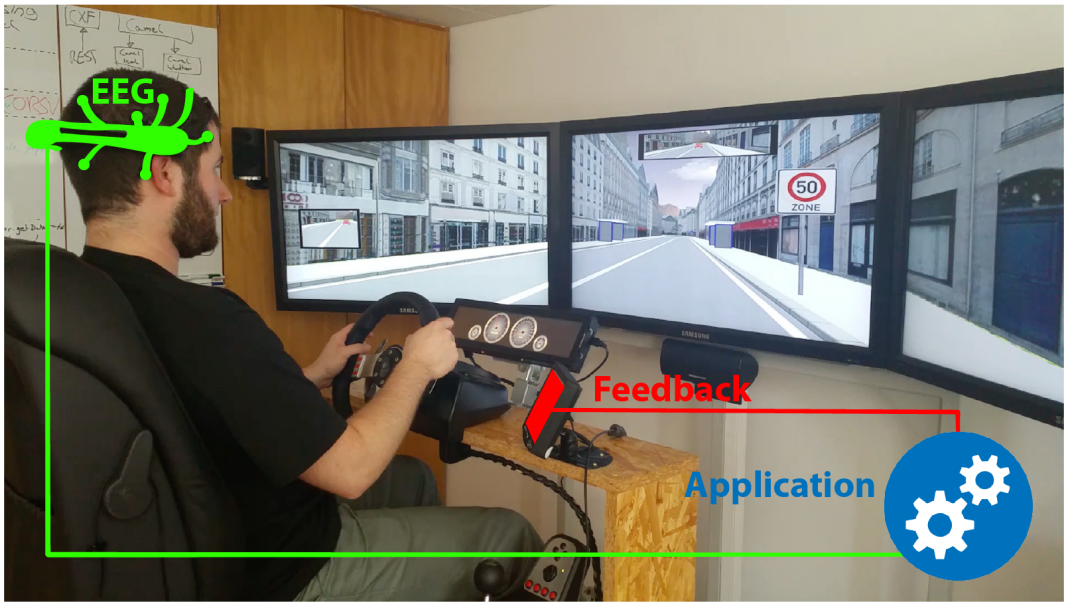

Abb. 1.1: Skizze des Systemaufbaus: Körpersensoren (Elektroenzephalografie / Elektrokardiogramm) liefert Daten an die Applikation und ein Feedback-Device warnt den müden Fahrer. Bild zeigt den Fahrsimulator der Reutlingen University.

den Serienbetrieb eignet, soll das System für die Validierung von berührungslosen Systemen verwendet werden. Die Notwendigen Testdaten werden im Rahmen des Projekts im Fahrsimulator der Reutlingen University aufgenommen. Dennoch soll ganze System leicht portierbar sein, um es in einem echten Fahrzeug testen zu können.

Die Ausarbeitung gliedert sich folgendermaßen. Im Kapitel 2 werden verschiedene Forschungsergebnisse zur Müdigkeitserkennung aufgezeigt und analysiert. Daraus ergeben sich die Anforderungen an eine Anwendung zur Müdigkeitserkennung. Die Infrastruktur, die Beschaffung der Daten und die durchgeführten Versuche sind Thema von Kapitel 4.2. Die Implementierung eines portablen Systems zur Müdigkeitserkennung mit Körpersensoren wird im Kapitel 5 vorgestellt. Kapitel 6 beschreibt die Ergebnisse und leitet in Kapitel 7 zu den weiteren Schritten und dem Fazit über. In folgenden Absatz werden Grundlagen für die kommenden Kapitel erläutert.

1.1 EEG Headset

Für das Projekt wurde ein EMOTIV Epoc+ EEG Headset [1] (Abb 1.3) verwendet. Es besitzt 14 EEG Kanäle, sowie ein Gyroskop und sendet seine Daten via Bluetooth an

[1] http://emotiv.com/epoc

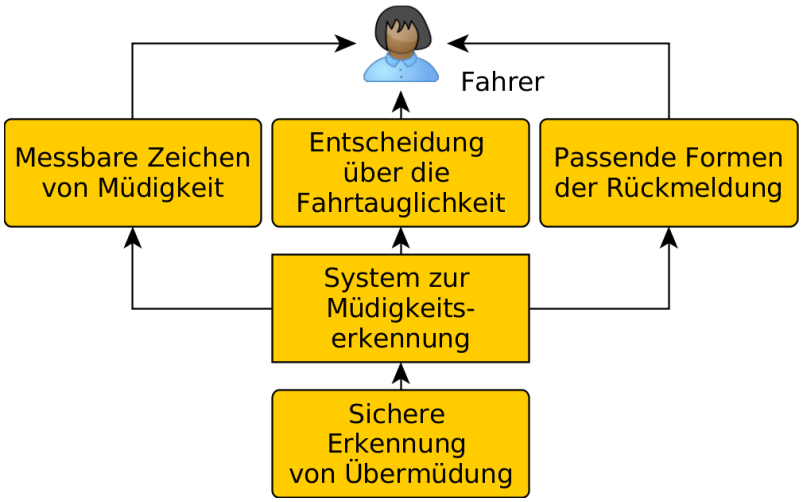

Abb. 1.2: Das System zur Müdigkeitserkennung gliedert sich in mehrere Teilproblemstellungen auf.

den Rechner. Das Headset wird über den Kopf gestülpt und besitzen an den Sensorenden einen Filz der mit Kochsalzlösung befeuchtet wird.

Abb. 1.3: Das EMOTIV Epoc+ EEG Headset wird einfach über den Kopf gestülpt.

Die Sensoren Anordnung ist an das internationale 10-20 System [6] angelehnt (1.4). Hierbei handelt es sich um eine standardisierte relative Anordnung der Elektroden. Die Rohdaten werden Abtastrate von 128 geliefert und enthalten neben dem Wert auch die Signalstärke (Qualität). Leider liefert die mitgelieferte SDK die Rohdaten nicht in Echtzeit, weshalb die Open-Source Lösung Emokit[2] eingebunden wurde.

[2] `https://github.com/openyou/emokit`

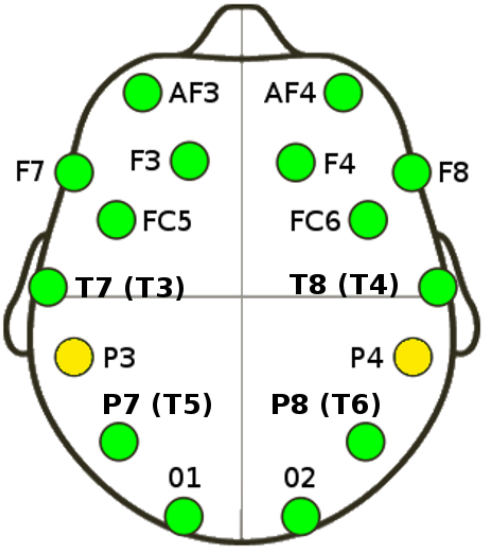

Abb. 1.4: Die 14 Kanäle (Grün), sowie die beiden Qualitätssensoren (Gelb).

2 Stand der Technik

Systeme zur Erkennung von Müdigkeit versuchen mit verschiedenen Parametern herauszufinden, ob sich die Person (der Fahrer) noch in einem aufmerksamen Zustand befindet. Die lässt sich in drei Bereiche unterteilen: Fahrverhalten, Computer-Vision (CV) und Körpersensoren.

In der Praxis setzen Automobilhersteller wie Daimler [3] und Volkswagen, sowie Automobilzulieferer wie Bosch [7] auf die Analyse des Fahrverhaltens. Insbesondere Spurhalten und ruckartiges Gegenlenken scheinen ein signifikantes Indiz für beginnende Übermüdung zu sein. Weiterhin sind externe Geräte und einige Apps für Smartphones erhältlich. Leider existieren kaum öffentlich zugängliche Arbeiten zu diesem Ansatz von Müdigkeitserkennung, da es sich um interne Entwicklungen handelt.

Beim CV-Ansatz wird der Fahrer und die Straße mit Hilfe von Kameras beobachtet. Zhang et al. [8] stellen hierzu eine Applikation mit der Microsoft Kinect vor. Es wurden sowohl die Kopfpose, als auch die Augenstatus bestimmt. Bergasa et al. [9] extrahierten aus dem Bild einer Infrarot Kamera mehrere Features, wie beispielsweise den prozentualen Anteil von geschlossenen Augen (Percent eye closure, PERCLOS). Mit dieser Technik erreichten sie bei der Erkennung von Übermüdung eine nahezu hundertprozentige Erfolgsrate. Kamerabasierte Systeme schränkenden Fahrer nicht ein, da kein direkter Kontakt zum Fahrer bestehen muss. Jedoch ist jede Kamera optischen Grenzen unterworfen (bspw. bei Nacht oder schlechtem Wetter).

Die beiden beschriebenen Bereich werden in dieser Arbeit nur für die manuelle Markierung der aufgenommen Datensätzen genutzt. Für die Erkennung von Müdigkeit werden verschiedene Körpersensoren bzw. deren Kombination (multimodal) eingesetzt. Meist werden elektrische Spannung am und im Körper gemessen. Neben dem EEG, werden bspw. die Elektrokardiographie (EKG) oder Elektrookulografie (EOG) genutzt. Beim EKG wird die elektrische Aktivität des Herzmuskels erfasst, um bspw. die Herzfrequenz zu bestimmen. Das EOG misst die Bewegung der Augen, um bspw. Blinzeln zu erkennen.

Ansätze mit einem EKG allein, zeigten in verschiedenen Arbeiten kein eindeutiges Ergebnis [10], [11]. Beide versuchten Informationen aus der Herzfrequenzvariabilität zu erhalten und diese zu klassifizieren. Khushaba et al. [12] versuchten

EEG, EKG und EOG zu verbinden und verglichen verschiedenen Kombinationen. Mit einem „fuzzy wavlet“ basierten Algorithmus wurde die Signale aufbereitet und zeigten, dass ein EEG alleine bereits ausreicht. Die Kombination eines EEG mit EKG bzw. EOG verbesserte das Ergebnis nicht signifikant. Auch Johnson et. al [13] kamen zu der Erkenntnis, dass ein EEG ausreicht und das genutzte EOG nicht benötigt wird. Subasi [14] konnte mit einem EEG die Zustände „Wach“, „Schläfrig“ und „Schlafend“ unterscheiden. Die Wavelet-Transformation und das genutzte künstliche Neuronale Netz (KNN) führten zu einer Erkennungsrate von 93%. Vuckovic et al. [15] fanden den besten Algorithmus für die Initialisierung des KNNs: Der Learning Vector Quantization Algorithmus. Im Vergleich mit EEG-Experten erreichte das KNN eine Übereinstimmung von 90%. Huang et al. [16] nutzten ein Hidden Markov Modell zu Erkennung und erreichten eine gute Erfolgsrate. Lin et al. [17] nutzten die Unabhängige Komponenten Analyse (UKA) und Lineare Regression (LR) und konnten zeigen, dass hiermit bis zu 88% richtige Ergebnisse erzielt werden können.

Die betrachteten Arbeiten unterstreichen die Eignung von Körpersensoren um Müdigkeit zu erkennen. Die Stärken im Bereich Präzision und Richtigkeit der Ergebnisse, im Vergleich zu Verhaltensanalyse oder CV-Techniken, zeigte sich in den Ergebnissen. In Sachen Komfort bleiben Körpersensoren jedoch hinter den anderen Ansätzen zurück. Verhaltensanalyse oder CV-Technik werden für die Arbeit nicht berücksichtigt und treten nur bei der Analyse / Klassifizierung der Testdaten in Erscheinung. Gähnen, häufiges blinzeln oder abkommen von der Fahrspur, können Hinweise auf Müdigkeit sein, sodass die entsprechenden EEG-Sequenzen gelabelt werden können. Aufgrund seiner höheren Genauigkeit, gegenüber EKG oder EOG, wird für die Anwendung ein EEG genutzt. Ein multimodales System ist vorbereitet, aber nicht umgesetzt. Die betrachteten Arbeiten zeigen in verschiedensten Ausführungen sehr gut Ergebnisse (~90% [17], [14]), sodass es sich beim EEG um eine aussichtsreiche Grundlage handelt. Bei der Klassifizierung lässt nutzen viele Ansätze ein Künstliches Neuronales Netz (KNN) [14] [15] [18] [19]. Weitere Ansätze nutzen die Lineare Diskriminanten Analyse [10] [12] oder ein SVM [20] [21]. Aufgrund der Häufigkeit in den anderen Arbeiten und der guten Bibliotheksunterstützung (PyBrain) wurde der Ansatz mit einem KNN weiter verfolgt. Die Vorgestellten Lösungen arbeiten mit medizinischen oder selbst gebauten EEGs. Alle Ansätze wurden in einer Simulierten Umgebung entwickelt und nie unter realen Bedingungen getestet. Aus der Analyse der verwandten Forschungsarbeiten ergeben sich Anforderungen an eine Anwendung zur Müdigkeitserkennung.

3 Anforderungen

Da es sich um ein sicherheitsrelevantes Fahrerassistenzsystem handelt, muss es in erster Linie präzise und korrekt funktionieren. Nicht ausgelöste Müdigkeitswarnungen wägen den Fahrer in Sicherheit, obwohl er evtl. nicht mehr in der Lage ist, sein Fahrzeug zu führen. Falsch ausgelöste Warnungen senken die Akzeptanz der Anwendung und führen im Extremfall zu deren Abschaltung. Zudem muss es robust gegen Störungen und Falscheingaben sein, es muss zu jeder Zeit gewährleistet sein, dass das System läuft bzw. den Fahrer im Fehlerfall rechtzeitig über den Status der Anwendung informieren.

Die Anwendung muss zudem nahezu in Echtzeit funktionieren und den Fahrer sofort über eine erkannte Müdigkeit informieren. Bei der der Implementierung muss auf die Performance der Erkennung geachtet werden. Eine zu späte Meldung an den Fahrer könnte zu einem Unfall führen. Um das System möglichst flexibel zu machen, sollte es auf verschiedenen Plattformen lauffähig sein, auch hier muss eine verringerte Rechenleistung beachtet werden (bspw. auf einem Smartphone).

Um die Software möglichst flexibel bzw. unabhängig von der Hardware zu machen, sollen Datenquelle und Datenverarbeitung möglichst lose gekoppelt sein und sich leicht auf verschiedenen Systemen ausführen lassen. Das hinzufügen von weiteren Quellen (bspw. EKG oder EOG) soll vorbereitet werden.

Die Rückmeldung der Anwendung soll den Fahrer warnen, sodass er diese auf jeden Fall wahrnimmt, jedoch nicht in die Fahrsituation eingreift. Es ist mehr als Hinweis zu verstehen und nicht als Maßregelung, da dies die Akzeptanz wiederum mindern könnte. Die Anwendung soll ohne lange Einrichtung oder Interaktion des Fahrers funktionieren.

Der Komfort beim Fahren sollte möglichst hoch sein und die Sensoren sollten den Fahrer möglichst wenig beeinträchtigen. Mit einem medizinischen EEG mit 64 Pins und vielen Kabeln, ist das kaum möglich. Das Emotiv EEG lässt sich wie eine Mütze aufsetzen und überträgt seine Daten via Bluetooth - ein Komfortgewinn. Im Produktiveinsatz ist das dennoch zu unbequem und wird in dieser Form nicht in Serie gehen können. Weiterhin soll das System unter realen Bedingungen getestet werden können und sich daher leicht vom Fahrsimulator der Reutlingen University

in ein echtes Auto oder einen anderen Simulator portieren lassen. So können Störungen während einer richtigen Autofahrt (die sich nicht simulieren lassen) erkannt werden. In einem anderen Simulator kann die Anwendung zur Validierung und Verbesserung von anderen Systeme verwendet werden.

Für ein Projekt an einer Hochschule, das auch für Folgeprojekte genutzt werden soll, ergeben sich ebenso Anforderungen an die Codequalität, insbesondere an Lesbarkeit und Wartbarkeit. Die Funktionalität und der Aufbau der Anwendung muss sauber dokumentiert werden, sodass der Einstieg für neue Entwickler möglichst reibungslos verläuft.

Diese Anforderungen sind in Tabelle 3.1 zusammengefasst. Wichtigste Punkte sind die Korrektheit, Portabilität und Komfort des Systems (Abb. 3.1). Wobei vor allem Portabilität und Komfort in den betrachteten Arbeiten bisher kaum berücksichtigt wurden.

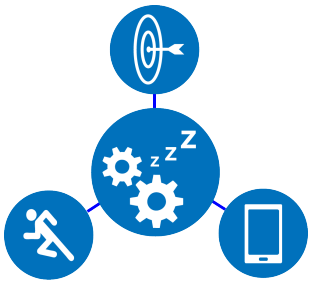

Abb. 3.1: Die Anwendung mit den Schwerpunkten: Korrektheit, Portabilität und Komfort

Tab. 3.1: Anforderungen

1	Die Anwendung muss Müdigkeit präzise und korrekt erkennen (wenige false positive und false negative)
2	Die Anwendung muss robust gegen Fehler / Fehleingaben sein
3	Die Anwendung muss den Fahrer über Störungen informieren
4	Die Anwendung muss den Fahrer so schnell wie möglich über eine erkannte Müdigkeit informieren
5	Die Anwendung muss sich in andere Umgebungen (Simulator oder echtes Fahrzeug) portieren lassen
6	Das eingesetzte EEG soll sich gut handhaben lassen und den Fahrer möglichst wenig beeinträchtigen
7	Die Anwendung soll auf möglichst vielen Plattformen lauffähig sein (Verschiedene Betriebssysteme und Gerätetypen)
8	Die Anwendung soll möglichst wenig Ressourcen verbrauchen (Siehe 5.)
9	Die Anwendung soll sich nahtlos in die Fahrsimulator-Umgebung einfügen
10	Datenakquise und Verarbeitung sollen auf getrennten Systemen durchgeführt werden können
11	Die Meldung über erkannte Müdigkeit soll vom Fahrer bemerkt werden, ihn aber nicht ablenken oder erschrecken
12	Die Anwendung (Module, Klassen, Methoden) soll komplett dokumentiert sein
13	Die Anwendung (Module, Klassen, Methoden) soll komplett durch Tests (mind. Unit-Tests) abgesichert sein

4 Rahmenbedingungen

Aus den Anforderungen des vorhergehenden Kapitels und der vorgegebenen Infrastruktur folgen die Rahmenbedingungen der Implementierung. Das Experiment zur Beschaffung der Testdaten wird dann im nächsten Abschnitt beschrieben.

4.1 Infrastruktur

Die Entwicklung der Anwendung (Experiment, Implementierung und Test) werden im Fahrsimulator des IoT Labs[1] der Reutlingen University (Abb. 4.1) durchgeführt. Der Fahrsimulator besteht aus einem Simulationsrechner mit drei 20"Monitoren, einem echten Fahrersitz, Lenkrad, Schaltung und Pedalen. Für die Simulation wird die Open-Source Software OpenDS[2] genutzt. Per TCP/IP werden alle notwendigen Fahrzeugdaten vom Simulationsrechner auf den Datensammler und dort in ein virtuelles Steuergerät-Software (Vector CANoe[3]) geschickt. Über einen CAN-Bus können die Daten dann wieder, mittels einer Schnittstelle (CAN-Interface), ausgelesen werden. Die eigentliche Applikation wird auf dem Embeddedrechner ausgeführt.

Die Anwendung selbst wird in Python 2.7 [4] realisiert. Python ist OpenSource, lässt sich leicht lernen und der Code ist, dank des ausgefeilten Sprachkonzepts, gut lesbar. Wie schon angemerkt, ist dies für Hochschulprojekte eine wichtige Eigenschaft, da es häufig zu Personalwechseln kommt. Für Python existieren eine Vielzahl an Bibliotheken für wissenschaftliche Anwendungen. Für das Projekt werden unter anderem die SciPy[5], welche sich an der Funktionalität von Matlab orientiert oder die MachineLearning-Bibliothek PyBrain[6] genutzt. Python läuft auf allen gängigen Betriebssystemen (Windows, Mac oder Linux) wenn der passende Python-Interpreter installiert ist. Weiterhin existieren Scripte, um eine Anwendung auch auf anderen Plattform lauffähig zu machen (Android, IPhone, etc.). Als Scriptsprache ist Python nicht ganz so schnell, als eine kompilierte Sprache (Java, C#), liefert jedoch ausreichende Performance. Weiterhin kann eine Python

[1] `http://iotlab.reutlingen-university.de/`
[2] `https://www.opends.eu`
[3] `https://vector.com/vi_canoe_de.html`
[4] `https://www.python.org`
[5] `http://www.scipy.org`
[6] `http://pybrain.org`

Anwendung in ByteCode kompiliert werden, um die Geschwindigkeit weiter zu steigern.

Die komplette Anwendung ist mit docstrings (Dokumentation im Code) versehen und über eine HTML Seite abrufbar. Weiterhin sind alle wichtigen Codestellen durch Unittests abgesichert. Um den Einstieg in manche Themengebiete zu erleichtern, sind einfach Beispiele und Visualisierungen implementiert. Der Komplette Code ist unter Versionskontrolle in einem git-Repository

Die Anwendung fügt sich nahtlos in die Simulationsumgebung ein, kann aber auch Standalone betrieben werden. Das CAN-Interface kann die EEG-Rohdaten direkt via http empfangen und auf den CAN-Bus legen, als wäre das EEG ein Fahrzeugsensor. Die EEG-Daten können von der Anwendung wieder vom CAN-Bus gelesen werden und dann entsprechend weiterverarbeitet werden. Der http-Server kann auch direkt angesprochen werden. Somit sind Datenbeschaffung (EEG-Rohdaten) und Verarbeitung getrennt und können auf unterschiedlichen Geräten ausgeführt werden.

4.2 Testdatenbeschaffung

Um Daten von übermüdeten Fahrern zu erhalten, kann natürlich kein Versuch im Straßenverkehr durchgeführt werden. Die Fremd- und Eigengefährdung wäre einfach zu groß. Darum fanden die Versuche in einer Simulationsumgebung statt. Für das Experiment wird im Fahrsimulator eine Nacht-Autobahnfahrt simuliert. Verschiedene Studien [22], [23] legen nahe, dass sich Simulationen zwar von der Realität unterscheiden, dass jedoch die Ergebnisse trotzdem valide und brauchbar sind.

Für den Versuch werden neben den rohen EEG Signalen, auch die Fahrzeugdaten, sowie die Simulation und der Fahrer aufgenommen. Weiterhin kann der Versuchsleiter auch Besonderheiten protokollieren (Abb. 4.2).

Im Versuch sollte der Fahrer eindeutige Anzeichen von Müdigkeit zeigen. Dies kann durch verschiedene Versuchsparameter begünstigt werden. So zeigt eine Studie, dass Unfälle meist zwischen 2:00 - 6:00, sowie 14:00 - 16:00 Uhr passieren [23]. Auch die Schlafmenge von weniger als 6 Stunden in der Nacht vor dem Experiment erhöht die Chance auf Anzeichen [22]. Das Geschlecht oder Alter der Probanden ist nicht relevant. Vor dem Experiment sollten jedoch keine Drogen, Alkohol oder

Kaffee eingenommen werden. Ein Führerschein ist von Vorteil, aber nicht zwingend notwendig.

Auch die Teststrecke (Abb. 4.3) trägt zur Erhöhung der Müdigkeit bei. Monotone Autobahnfahrten die größtenteils geradeaus verlaufen, ohne andere Verkehrsteilnehmer und konstanter Geschwindigkeit führen eher zu einer Übermüdung. Nach diesen Kriterien wurde eine endlose zweispurige Autobahnkarte mit einer Geschwindigkeit von konstant 130Kmh erstellt. Sie spielt zudem Nachts und ist eher Dunkel gehalten, was besonders anstrengend für die Augen ist.

Für einen Versuch werden 40 Minuten angesetzt. Fünf Minuten werden für eine kurze Einführung, 30 Minuten für die Testfahrt und wieder fünf Minuten für eine kurze Einschätzung mit Fragebogen.

Anhand der aufgenommenen Daten werden nun Stellen gesucht, an denen die Testperson eindeutige Anzeichen von Müdigkeit zeigt. Eindeutig sind Verhaltensweisen wie häufiges Gähnen und Einnicken (Kopf fällt nach vorn) - Diese Merkmale werden häufig in CV-Ansätzen genutzt. Auch Verhaltensmerkmale wie, von der Spur abkommen und heftig Gegenlenken oder deutliche Veränderungen der Geschwindigkeit können Anzeichen für eine Unachtsamkeit wegen Müdigkeit sein. Diese Stellen werden dann in den EEG Daten mit dem Label „Müde“ markiert, alle anderen mit „Wach“. Die EEG Sequenzen können dann auf eindeutige Varianzen untersucht werden. Im nächsten Kapitel ist dies Thema der Datenaufbereitung.

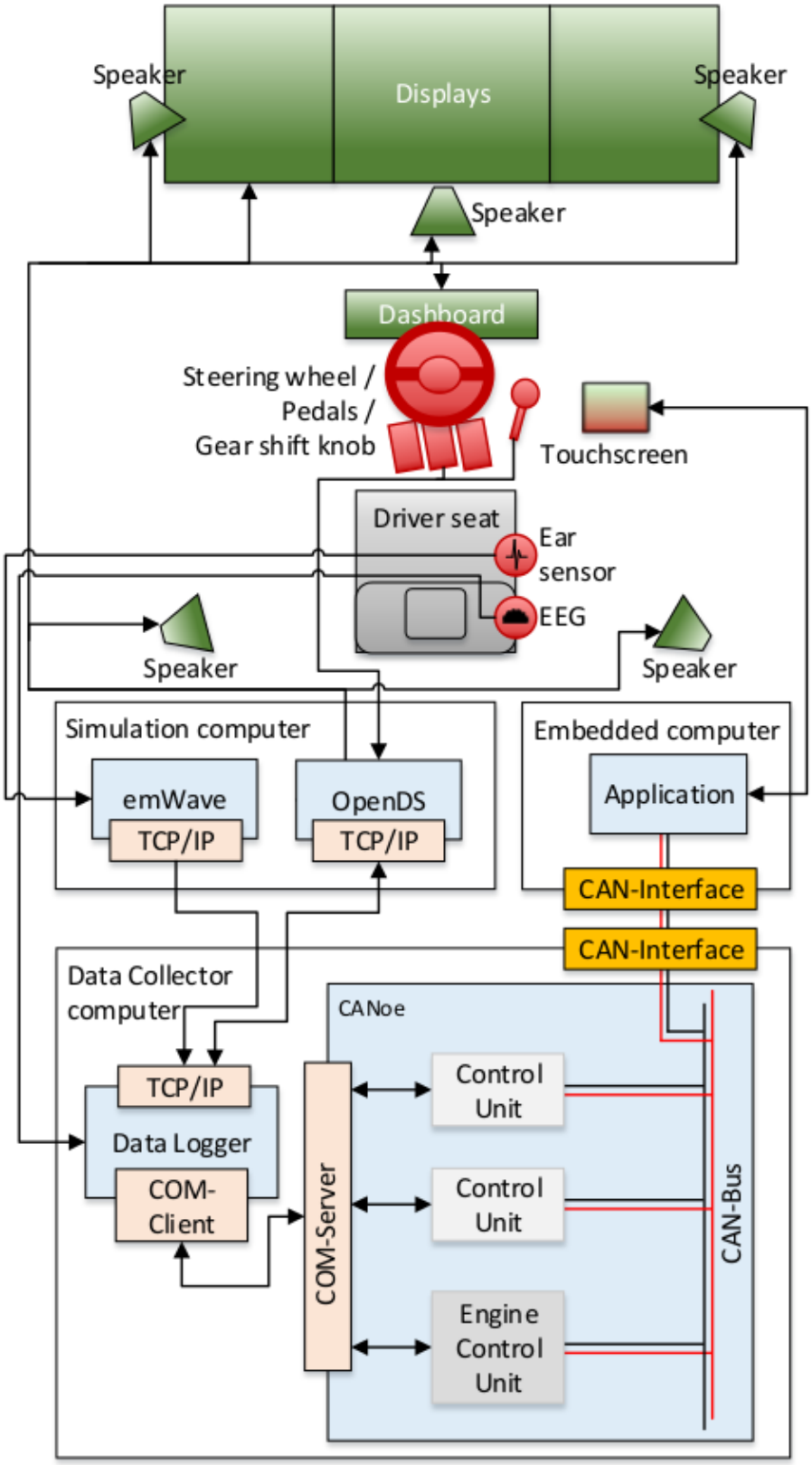

Abb. 4.1: Der Aufbau des Simulators der Reutlingen University mit den drei Rechnern für die Simulation, Datensammlung und Applikation.

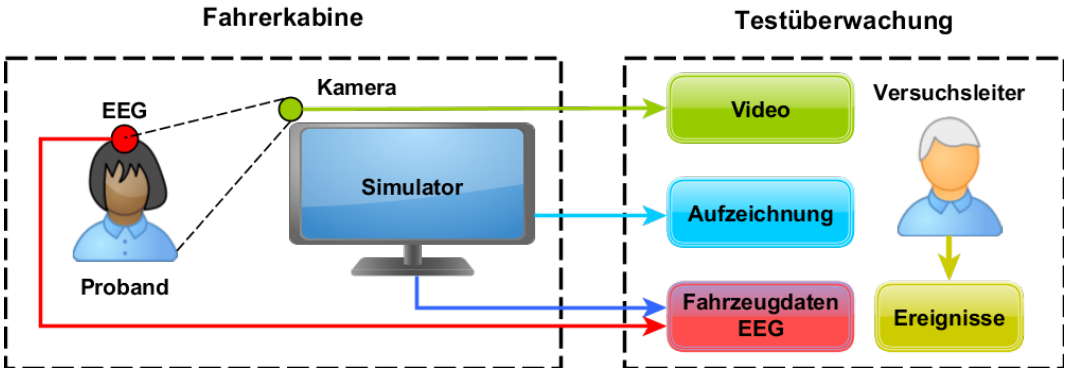

Abb. 4.2: Der Versuchsaufbau für die Erkennung von Müdigkeit mit den Datenströmen aus der Fahrerkabine zur Testüberwachung.

Abb. 4.3: Die Autobahnkarte für den Versuch verläuft endlos geradeaus, Nachts und bei konstanter Geschwindigkeit.

5 Müdigkeitserkennung mit einem EEG-Headset

Abbildung 5.1 zeigt die Gesamtübersicht des entwickelten Systems dar. Alle blauen Klassen der Anwendung wurden in Python 2.7 implementiert. Der Fahrsimulator und seine Schnittstellen (rosa) in Java und C#, die einzelnen Datenströme sind Grün gekennzeichnet. Bedingte Abzweigungen für alternative Wege in Gelb. Die Anwendung läuft verteilt auf dem DataCollector und dem Embedded PC. Die Datenquelle befindet sich auf dem DataCollector, von dort werden die Daten via CAN-Bus an den Embedded PC übertragen und verarbeitet (Abb. 5.2).

Die Daten werden über mehrere Threads übertragen, an unkritischen Stellen passiert dies per direktem Zugriff oder einem Callback (Abb. 5.3). Derzeit sind 2 SignalWindows implementiert, diese Zahl lässt sich erweitern. Die Abarbeitung der ProcessingChain ist potentiell aufwändig und ist mit zwei Thread-sicheren Queues umgesetzt. Sollte die Verarbeitung zu lag dauern, könnten weitere ProcessingChains hinzugefügt werden. Der FeatureExtractor reicht diese Daten derzeit auf einer weiteren Queue zur Hauptanwendung, wo die Klassifizierung blockierend angestoßen wird und das Ergebnis zum Ausgabebildschirm geleitet wird. Der FeatureExtractor kann zu einem späteren Zeitpunkt eine Aggregation der Daten vornehmen oder sich um die Verwaltung der ProcessingChains kümmern.

Abschnitt 5.1 befasst sich mit den Rohdaten und deren Weiterreichung im System. Die Verarbeitung der Rohdaten ist Thema von Abschnitt 5.2. Im folgenden Abschnitt werden daraus die passenden Merkmale extrahiert. In Abschnitt 5.4 wird die Arbeitsweise des Klassifikators näher beleuchtet.

5.1 Datensammlung

Das EEG Headset schickt enkodierte Byte-Sequenzen via Bluetooth, an die proprietären Emotiv Premium Libraries oder die Open-Source Lösung Emotkit (vgl. 1.1). Die Emokit Klasse wurde für das Projekt leicht modifiziert, sodass sie sich ins System einfügt. So wurde Unterstützung für das neueste EPOC+ Modelle implementiert, sowie die Möglichkeit, Testdaten aus aus dem TableReader zu versenden. Die dekodierten Rohdaten enthalten 14 EEG Kanäle mit Wert und Qualität, die

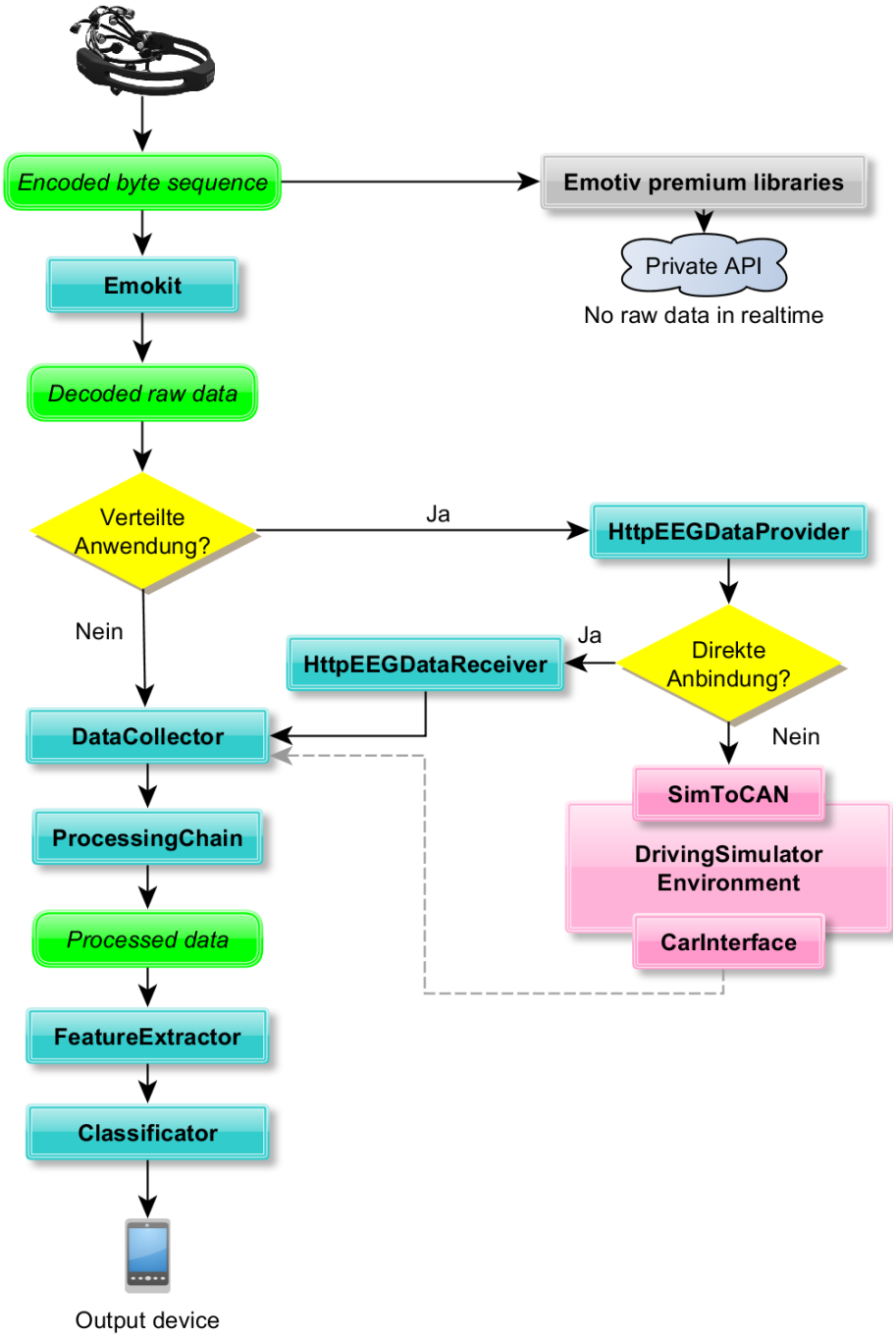

Abb. 5.1: Der Aufbau des entwickelten System zur Müdigkeitserkennung. Grün: Datenströme, Blau: Python Klassen der Anwendung, Gelb: bedingte Abzweigungen, Rosa: Klassen der Fahrsimulatorumgebung

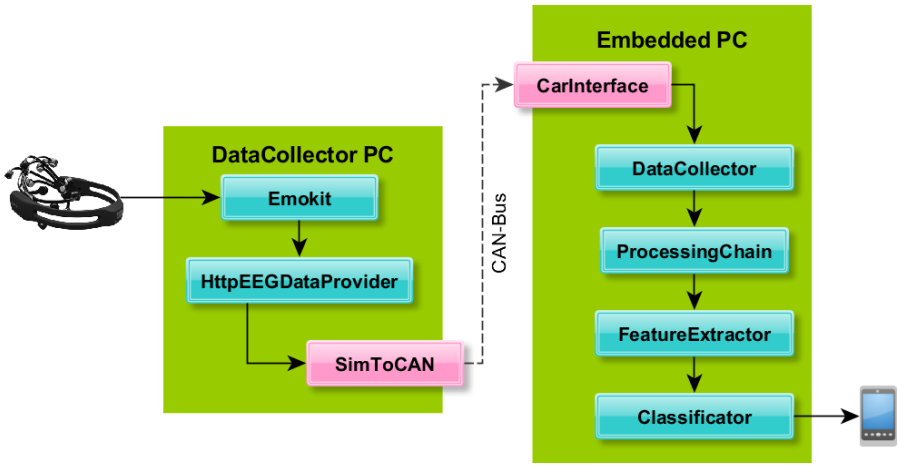

Abb. 5.2: Datenquelle und Verarbeitung sind verteilt im Fahrsimulator eingebettet. Die Übertrageung erfolgt via CAN-Bus.

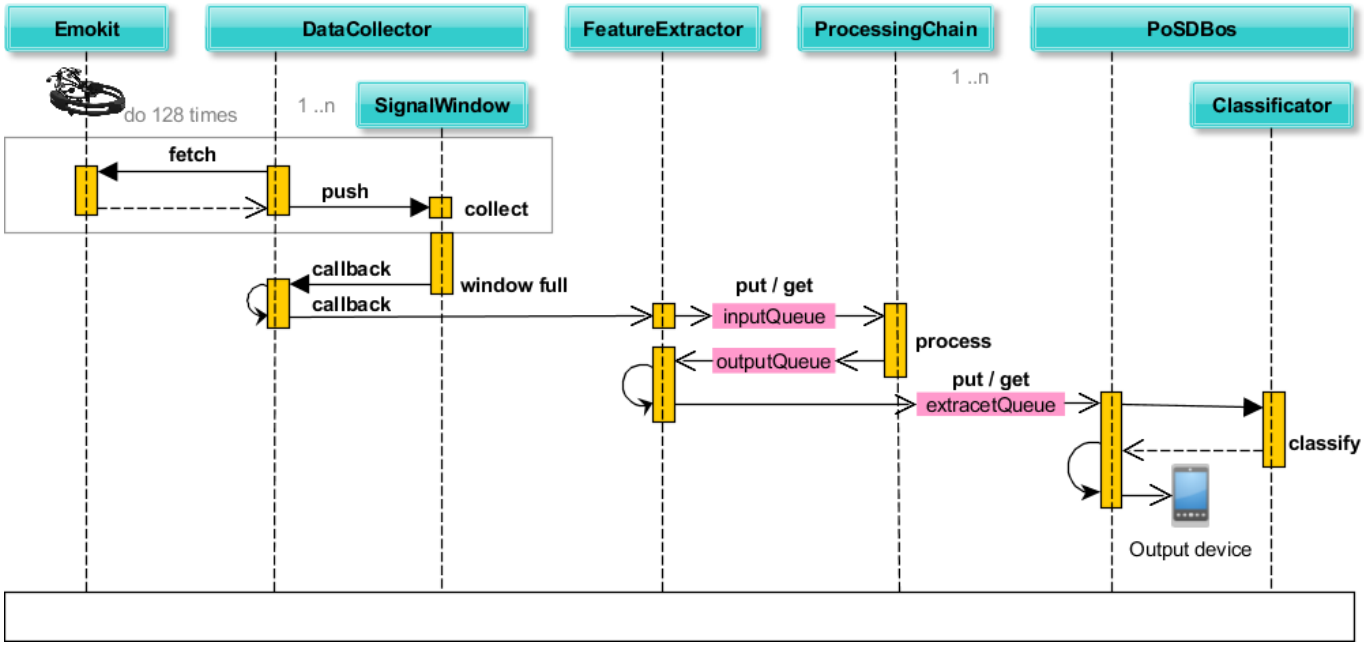

Abb. 5.3: Die Anwedung ist in mehrere Threads unterteilt. SignalWindow und Processing Chain können mehrere Instanzen haben.

Gyroskopwerte in X- und Y-Richtung und einen Zeitstempel (Abb. 5.4) Mit einer Abtastrate von 128Hz sind das 128 * 16 = 2048 Werte pro Sekunde. Diese Daten können als CSV gespeichert werden, sowie an einen HTTP-Server oder direkt an den DataCollector übergeben werden. Über den Server integriert die SimTo-CAN Anwendung via HTTP die EEG Daten in den Fahrsimulator und das virtuelle Steuergerät.

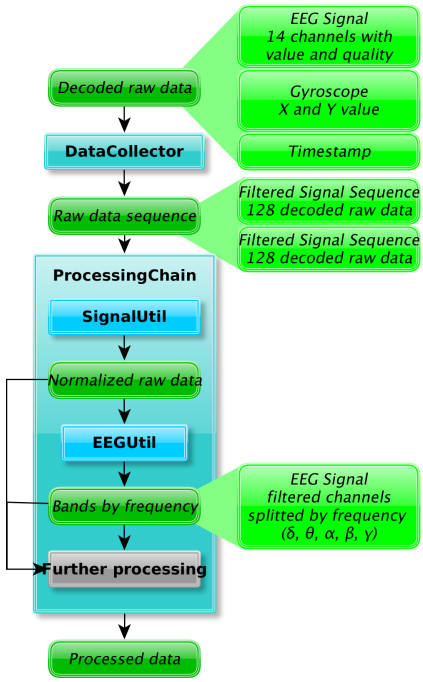

Abb. 5.4: Die Daten werden immer weiter reduziert und verarbeitet.

Der DataCollector kann Daten direkt oder aus einem HTTP-Client empfangen. Die Schnittstelle zum Fahrsimulator bzw. zum CAR-Interface ist vorbereitet. So ist es möglich die Datensammlung und die Datenverarbeitung auf verschiedenen Rechnern durchzuführen.

Die Aufgabe der DataCollector-Klasse ist es, die einzelnen Signale in Sequenzfenstern von 128 Signalwerten zu aggregieren. Das ist die Abtastrate des EEG-Headsets und entspricht somit in etwa den Signalen einer Sekunde. Es sind zwei dieser Fenster implementiert und sie überschneiden sich in der Hälfte. So ist gewährleistet, dass signifikante Stellen nicht verloren gehen. Die Fensterfunktion ist ein simples Rechteck, sodass alle Werte gleich gewichtet werden. Eine andere Fensterfunktion bspw. mit Glockenkurvenerlauf (Hamming- oder Hann-Fenster) wäre einfach einzubauen. Auf den ersten Blick ist die Fensterlösung eine Verdopplung der Daten, jedoch fügt der DataCollector nur konfigurierte Kanäle hinzu, sodass sich die Datenmenge wieder deutlich reduzieren lässt. Im folgenden Abschnitt werden die gefilterten Sequenzfenster nun verarbeitet und aufbereitet.

5.2 Datenverarbeitung

Die gefilterten EEG-Sequenzen durchlaufen nun eine Verarbeitungskette (Abb. 5.4). Im ersten Schritt werden die Signale auf das Intervall 1 bis -1 normalisiert , dazu werden die Signale durch den jeweiligen Maximalwert bzw. Betrag des Minimalwertes geteilt. Dieses Vorgehen stellt sicher, dass die absolute Amplitude keinen Einfluss auf die Gewichtung im Klassifikator hat. Zudem wird die Datenmenge wiederum reduziert. Im zweiten Schritt werden die Signale in Frequenzbänder (EEG-Bänder) unterteilt. Hierzu werden bestimmte Frequenzbereiche aus dem Signal entfernt, sodass nur die gewünschten Frequenzen erhalten bleiben. Diese EEG-Bänder gliedern sich in folgende Frequenzbereiche und werden nach griechischen Buchstaben benannt:

- δ : 0,1 bis $<$ 4Hz
- θ : 4 bis $<$ 8Hz
- α : 8 bis $<$ 13Hz
- β : 13 bis $<$ 30Hz
- γ : $>$ 30Hz

Den Frequenzbändern werden verschiedene Eigenschaften zugesprochen. Delta Wellen treten bei Erwachsenen häufig in der traumlosen Tiefschlafphase auf. Theta-Wellen zeigen sich bei Schläfrigkeit und leichtem Schlaf. Mit leichter Entspannung und entspannter Wachheit (mit geschlossenen Augen) werden Alpha-Wellen assoziiert. Beta Wellen treten während der REM-Schlafphase oder unter Einwirkung von Psychopharmaka auf. Gamma-Wellen gehen häufig mit starker Konzentration oder Meditation einher.

Um die einzelnen Frequenzbänder zu erhalten ist eine Filterfunktion notwendig. Für die Anwendung wurde hierfür ein Buttworth-Filter [24] eingesetzt. Gleichung 5.1 zeigt die Übertragungsfunktion mit A_0: Gleichspannungsverstärkung, $\Omega = \frac{f}{f_g}$: auf Grundfrequenz normierte Frequenz und n: Ordnung des Filters. Abbildung 5.5 zeigt exemplarisch Filterfunktionen verschiedener Ordnung mit den Grenzen von 500 bis 1250Hz. Alle Frequenzen darunter und darüber werden deutlich abgeschwächt bzw. gehen gegen null. Der Butterworth-Filter verläuft nahe Eins im gewünschten Bereich, fällt an den Grenzen ab und stellt sicher, dass das Signal an den Grenzen um $\frac{1}{\sqrt{2}} \approx 0.7071$ gemindert wird. Je höher die Ordnung, desto steiler geht die Funktion durch die angegebenen Grenzen. Der Filter lässt sich gut in Hardware realisieren.

$$|\underline{A}|^2 = \frac{A_0^2}{1 + k_{2n}\Omega^{2n}} \tag{5.1}$$

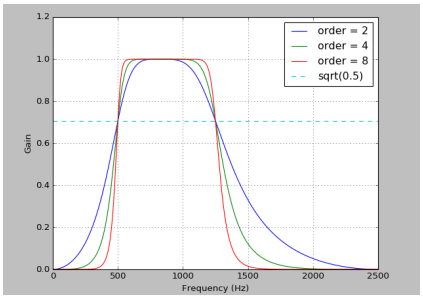

Abb. 5.5: Die Filterfunktion eines Butterworthfilters 2., 4,. und 8. Ordnung im Frequenzbereich von 500 bis 1250Hz.

5.3 Mermalsextraktion

TODO

5.4 Klassifikator

Nachdem in vorherigen Abschnitt eine passende Merkmalsmenge erstellt wurde, geht es nun um die Entscheidung, ob der Fahrer Müde oder Wach ist bzw. ob das System eine Müdigkeitsmeldung erscheinen lässt. Für diese Klassifizierung werden im allgemeinen Machine-Learning-Algorithmen verwendet. Anhand von markierten Datensätzen wird versucht den Algorithmus zu Trainieren (Überwachtes Lernen). Dies dient dem Ziel, dass er auch unbekannte Daten klassifizieren kann. Dieser Vorgang wird Generalisierung bezeichnet und ist auch im menschlichen Lernen ein wichtiger Schritt.

Für die Anwendung wurde zur Klassifizierung ein künstliches Neuronales Netz (KNN) ausgewählt. Es basiert auf einem erweiterten McCulloch-Pitts-Neuron [25] und ist der Funktionsweise des menschlichen Gehirns bzw. seinen Neuronen nachempfunden [26]. Ein KNN lässt sich im einfachste Fall durch eine Merkmalsmenge $X = x_1, x_2 ... x_n$, dazugehörige Gewichte $W = w_1, w_2 ... w_n$, eine Übertragungsfunktion $\sum$ und eine Schwellwertfunktion θ beschreiben (Abb. 5.6).

Dieser vereinfachte Aufbau kann schon einfach Aufgaben, wie bspw. ein logisches „UND“, erfüllen. Jedoch lässt sich schon ein logisches „XOR“ nicht mehr abbilden. Dafür müssen weitere Schichten von Neuronen (Hidden Layers) hintereinander geschaltet werden - das sog. Multi Layer Perceptron (MPL, Abb. 5.7).

Es exisitiert kein bekannter Algorithmus für die Wahl der optimalen initialen Parameter eines KNNs. Vuckovic et al. [15] hatten sich mit diesem Thema genauer beschäftigt und die Ergebnisse werden für die Versuche herangezogen. **TODO**

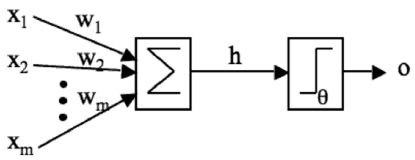

Abb. 5.6: Darstellung eines McCulloch-Pitts-Neurons. Die Merkmale X werden mit den Gewichten W multiplziert und in $\sum$ summiert. Wenn $h > \theta$ „feuert“ das Neuron $(o = 1)$ [26].

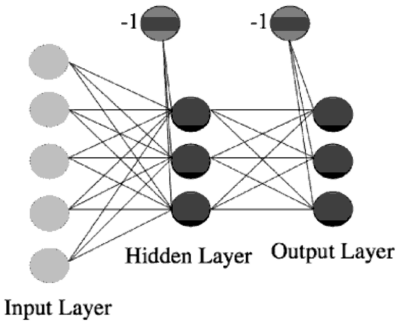

Abb. 5.7: Darstellung eines Neuronalen Netzes mit mehreren Schichten (Multi Layer Perceptron, MLP) [26].

6 Ergebnis

TODO Da das EEG in den letzte 4 Monaten in Reparatur und somit nicht verfügbar war, sind die Ergebnisse bisher nicht aussagekräftig. Weder das Experiment, noch die Merkmalsextraktion und die Klassifizierung konnten sinnvoll evaluiert werden.

Teil eins (1-4) der Anforderungen (vgl. Tab 3.1) kann erst nach erfolgreicher Implementierung getestet werden. Die Portierung ist theoretisch sehr einfach möglich, wenn während der Fahrt ein Laptop genutzt wird (5). Die Handhabung und Komfort des Headsets ist im Vergleich zu medizinischen EEGs deutlich verbessert (6). Wie schon angenommen, ist es jedoch für den Produktiveinsatz ungeeignet. Das EEG lässt sich ähnlich wie eine Mütze tragen und ist sehr leicht, sodass man schnell vergisst, dass es da ist. Die kabellose Übertragung sorgt für maximale Beweglichkeit.

Tests auf einem Smartphone oder Tablet müssen gesondert erfolgen, da es unter anderem von den Treibern des EEG Headsets abhängt (7). Auch Lasttests sind erst nach fertiger Implementierung wirklich aussagekräftig (8). Der Einbau in die Simulationsumgebung ist bis zum eintragen der Werte im CAN-Bus umgesetzt. Das Herausnehmen der Werte ist noch nicht vollständig umgesetzt, da die Integration der CarInterface Anwendung nicht funktionierte (9). Die Anwendung lässt sich sehr gut auf verteilten Systemen ausführen, auch über den Anwendungsfall des Fahrsimulators hinaus. Die akquirierten Daten können via http an die Verarbeitungsschicht übertragen werden. Auch das Verschicken des Ergebnisses der Klassifizierung per http wäre denkbar und einfach umzusetzen (10). Die Art der Benachrichtigung über eine erkannte Müdigkeit ist derzeit noch nicht vollständig umgesetzt. Dem Fahrer wird entweder ein grüner oder roter Bildschirm angezeigt. Ob dies die passende Form ist, bleibt zu klären.

Die komplette Anwendung ist mit Docstring[1] versehen, aus denen eine html-Dokumentation erzeugt werden kann. Weiterhin sind schwierige Code-Teile mit einfachen Beispielen erweitert, um den Einstieg zu erleichtern. Ob es so möglich ist, als Neuling, selbständig die Anwendung zu verstehen bzw. zu erweitern hängt wohl auch von der jeweiligen Person ab. Werden jedoch Veränderungen vorgenommen, kann durch die Unit-Tests sichergestellt werden, dass die Klassen weiterhin das Richtige tun. Integrationstests wären eine sinnvolle Erweiterung.

[1] `https://www.python.org/dev/peps/pep-0257/#what-is-a-docstring`

7 Fazit und Ausblick

TODO

8 Master Projekt

Neben der Entwicklung der Anwendung zur Müdigkeitserkennung, standen einige Termine für das IoT-Lab an. Unter anderem war das Thema auf der WVK.15[1], der InformaticsInside [2], am Tag der offenen Tür und dem Studientag präsent. Die in diesem Dokument genutzte LaTeX-Vorlage für die Ausarbeitung[3], wurde an den Stand der vorgegebenen Word-Vorlage angepasst und zur Verfügung gestellt.

Die Einarbeitung und vor allem die Dokumentation des Fahrsimulators war eine weitere wichtige Aufgabe. Viele Fragen waren nach dem Weggang von Emre Yay nicht abschließend geklärt. Es war also viel Suchen, Debuggen und Nachfragen notwendig. Um anderen diesen Aufwand zu ersparen war es wichtig den Simulator zu dokumentieren. Dazu musste zuerst das Format entschieden werden, da mehrere Personen, am besten auch parallel, an der Dokumentation arbeiten sollten. Ein Wiki[4] in der Umgebung der der IoT-Website erschien am besten für diese Aufgabe. Die Struktur und erster Inhalt waren die ersten Aufgaben für die Dokumentation. Weiterhin sind erste Schritte dokumentiert, sodass ein Neuling einen Einstiegspunkt für einfache Aufgaben (bspw. Simulation starten) hat. Juniors im Masterprojekt konnten bereits von einer Einführung profitieren.

Während der Entwicklung gab es immer wieder Kommunikationsbedarf mit verschiedenen Supportstellen. So traten Probleme mit OpenDS, CANoe und Emokit auf, die mit dem jeweiligen Support per Email oder Telefon geklärt werden mussten. Den meisten Aufwand erzeugt in dieser Hinsicht das einschicken des EEG Headsets. Nachdem der Defekt eines Sensors Anfang Februar festgestellt wurde, war nach einigen Emails mit dem Emotiv Support und in Absprache mit MKI Service klar, dass das Headset eingeschickt werden muss. Dies passiert dann Anfang März und verzögerte sich nochmal wegen Problemen am Zoll. Ende April kam das Headeset dann im TechCenter der Firma an und mehrere Nachfragen lieferten keine neuen Erkenntnisse. Ende Mai kam die Nachricht, dass der Fehler gefunden und behoben wäre. Da dieser Fall jedoch vorher getestet wurde, wurde dies Emotiv mitgeteilt mit der Bitte noch einmal zu testen. Beim zweiten Test wurde dann

[1] `http://wvk.reutlingen-university.de/index.php?site=topic&id=150299`
[2] `http://www.infoinside.reutlingen-university.de/index.php?site=program`
[3] `https://relax.reutlingen-university.de/course/view.php?id=6884`
[4] `http://iotlab.reutlingen-university.de/iotlab_wiki/index.php/Driving_Simulator`

festgestellt, dass das Headset nicht stabil läuft und es wohl ein anderes Problem war. Nach weiteren 3 Wochen entschied man sich, ein neues Gerät zu verschicken. Weitere Nachfragen brachten dann auch die Herausgabe eines Tracking-Codes und das EEG ist auf dem Weg. Die Notwendigkeit zu häufigen Nachfragen und die Reaktionszeiten von jeweils mindestens einer Woche, waren frustrierend und zeitraubend.

9 Literaturverzeichnis

[1] Strategy Analytics. Advanced driver assistance systems forecast - aug 2015. https://www.strategyanalytics.com/access-services/automotive/powertrain-body-chassis-and-safety/market-data/report-detail/advanced-driver-assistance-systems-forecast---aug-2015, 2015. Zugriff: 2015-10-28.

[2] Xavier Mosquet, Michelle Andersen, and Aakash Arora. A roadmap to safer driving through advanced driver assistance systems. https://www.bcgperspectives.com/Images/MEMA-BCG-A-Roadmap-to-Safer-Driving-Sep-2015.pdf, 2015. Zugriff: 2015-10-28.

[3] Daimler AG. Attention assist, 2008. Available at http://media.daimler.com/dcmedia/0-921-658892-49-1147698-1-0-0-0-0-1-11702-854934-0-1-0-0-0-0-0.html, Zugriff: 2015-08-13.

[4] Statistisches Bundesamt. Zahl der verkehrstoten steigt im jahr 2015 vorrausichtlich auf etwas 3450. https://www.destatis.de/DE/PresseService/Presse/Pressemitteilungen/2015/12/PD15_463_46241.html, 2015. Zugriff: 2015-12-31.

[5] Claudia Evers. Unterschätzte Risikofaktoren Übermüdung und ablenkung als ursachen für schwere lkw-unfälle. http://www.dvr.de/presse/seminare/904_20.htm, 2008. Zugriff: 2015-10-20.

[6] H. H. Jasper. The ten twenty electrode system of the international federation. *Electroencephalography and Clinical Neurophysiology*, 10:371–375, 1958.

[7] Robert Bosch GmbH. Bosch driver drowsiness detection, 2012. Available at http://www.bosch-presse.de/presseforum/details.htm?txtID=5037, Zugriff: 2015-08-13.

[8] Liyan Zhang, Fan Liu, and Jinhui Tang. Real-time system for driver fatigue detection by rgb-d camera. *ACM Trans. Intell. Syst. Technol.*, 6(2):22:1–22:17, March 2015.

[9] Luis M. Bergasa, Jesus Nuevo, Miguel A. Sotelo, Rafael Barea, and Maria E. Lopez. Real-time system for monitoring driver vigilance. *Intelligent Transportation Systems, IEEE Transactions on*, 7(1):63–77, March 2006.

[10] Jose Vicente, Pablo Laguna, Ariadna Bartra, and Raquel Bailon. Detection of driver's drowsiness by means of hrv analysis. In *Computing in Cardiology, 2011*, pages 89–92, Sept 2011.

[11] E. Rogado, J.L. Garcia, Rafael Barea, Luis M. Bergasa, and Elena Lopez. Driver fatigue detection system. In *Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on*, pages 1105–1110, Feb 2009.

[12] Rami N. Khushaba, Sarath Kodagoda, Sara Lal, and Gamini Dissanayake. Driver drowsiness classification using fuzzy wavelet-packet-based feature-extraction algorithm. *Biomedical Engineering, IEEE Transactions on*, 58(1):121–131, Jan 2011.

[13] Robin R. Johnson, Djordje P. Popovic, Richard E. Olmstead, Maja Stikic, Daniel J. Levendowski, and Chris Berka. Drowsiness/alertness algorithm development and validation using synchronized EEG and cognitive performance to individualize a generalized model. *Biological psychology*, 87(2):241–250, May 2011.

[14] Abdulhamit Subasi. Automatic recognition of alertness level from eeg by using neural network and wavelet coefficients. *Expert Syst. Appl.*, 28(4):701–711, May 2005.

[15] Aleksandra Vuckovic, Vlada Radivojevic, Andrew C.N. Chen, and Dejan Popovic. Automatic recognition of alertness and drowsiness from {EEG} by an artificial neural network. *Medical Engineering & Physics*, 24(5):349 – 360, 2002.

[16] Ruey S. Huang, Chung .J. Kuo, Ling-Ling Tsai, and Oscal T.C. Chen. Eeg pattern recognition-arousal states detection and classification. In *Neural Networks, 1996., IEEE International Conference on*, volume 2, pages 641–646 vol.2, Jun 1996.

[17] Chin teng Lin, Ruei cheng Wu, Sheng fu Liang, Wen hung Chao, Yu jie Chen, and Tzyy ping Jung. Eeg-based drowsiness estimation for safety driving using independent component analysis. *IEEE Trans. Circuits Syst. I, Reg. Papers*, pages 2726–2738, 2005.

[18] Beth J. Wilson and Thomas D. Bracewell. Alertness monitor using neural networks for eeg analysis. In *Neural Networks for Signal Processing X, 2000. Proceedings of the 2000 IEEE Signal Processing Society Workshop*, volume 2, pages 814–820 vol.2, 2000.

[19] Khaled B. Khalifa, Mohamed H. Bedoui, R. Raytchev, and Mohamed Dogui. A portable device for alertness detection. In *Microtechnologies in Medicine and Biology, 1st Annual International, Conference On. 2000*, pages 584–586, 2000.

[20] Hanbit Park, Seungwon Oh, and Minsoo Hahn. Drowsy driving detection based on human pulse wave by photoplethysmography signal processing. In *Proceedings of the 3rd International Universal Communication Symposium*, IUCS '09, pages 89–92, New York, NY, USA, 2009. ACM.

[21] Aihua Zhang and Fenghua Liu. Drowsiness detection based on wavelet analysis of ecg and pulse signals. In *Biomedical Engineering and Informatics (BMEI), 2012 5th International Conference on*, pages 491–495, Oct 2012.

[22] Johan Engstrom, Emma Johansson, and Joakim Ostlund. Effects of visual and cognitive load in real and simulated motorway driving. *Transportation Research Part F: Traffic Psychology and Behaviour*, 8(2):97–120, March 2005.

[23] Jim Horne and Louise Reyner. Vehicle accidents related to sleep: a review. *Occupational and Environmental Medicine*, pages 289–294, May 1999.

[24] Stephen Butterworth. On the theory of filter amplifiers. *Wireless Engineer*, 7, 1930.

[25] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[26] Stephen Marsland. *Machine learning : an algorithmic perspective*. Chapman & Hall/CRC machine learning & pattern recognition series. CRC Press, Boca Raton, 2009. A Chapman & Hall book.

10 Anhang

11 Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich die vorliegende Arbeit selbstständig erstellt, keine nicht genannte fremde Hilfe in Anspruch genommen und alle von mir verwendeten Hilfsmittel und Quellen in der Arbeit benannt und kenntlich gemacht habe.

Mir ist bekannt, dass eine unwahre Erklärung als Täuschung gewertet wird.

_____	_____	_____
Ort,	Datum	Unterschrift (Vor- und Nachname)