Maximilian Eibl, Martin Gaedke (Hrsg.)

Studierendensymposium Informatik 2016 der TU Chemnitz

Maximilian Eibl, Martin Gaedke (Hrsg.)

Studierendensymposium Informatik 2016 der TU Chemnitz

TECHNISCHE UNIVERSITÄT
CHEMNITZ

Universitätsverlag Chemnitz
2016

Impressum

Bibliografische Information der Deutschen Nationalbibliothek

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Angaben sind im Internet über http://dnb.d-nb.de abrufbar.

Cover-Foto: TU Chemnitz/Wolfgang Thieme

Technische Universität Chemnitz/Universitätsbibliothek
Universitätsverlag Chemnitz
09107 Chemnitz
http://www.tu-chemnitz.de/ub/univerlag

Herstellung und Auslieferung
Verlagshaus Monsenstein und Vannerdat OHG
Am Hawerkamp 31
48155 Münster
http://www.mv-verlag.de

ISBN 978-3-944640-85-3

http://nbn-resolving.de/urn:nbn:de:bsz:ch1-qucosa-201104

Vorwort zum Tagungsband

Die Technische Universität feiert 2016 ihr 180jähriges Bestehen. Begangen wird dieses Jubiläum mit einer Festwoche. Wir nehmen das Jubiläum zum Anlass, um das zweite Studierendensymposium der Fakultät Informatik der TU Chemnitz durchzuführen und freuen uns, die Ergebnisse in diesem Tagungsband vorlegen zu können.

Das Studierendensymposium Informatik ist themenoffen und richtet sich inhaltlich an alle Themen rund um die Informatik und ihre Anwendungen: Ob Hardware oder Software, ob technische Lösungen oder Anwenderstudien, ob Programmierung oder Verwendung, ob Hardcore-Technik oder gesellschaftliche Fragestellungen – alles, was mit informatischen Lösungen zu tun hat, ist willkommen. Das Studierendensymposium Informatik ist dabei weder auf die Fakultät Informatik noch auf die TU Chemnitz begrenzt. Wir haben explizit Einreichungen aus thematisch angrenzenden Fächern beworben und Hochschulen der Region in die Planung und Organisation eingebunden.

Studierende haben hier eine Plattform, um Projekte, Seminar- oder Studienarbeiten, Forschungsvorhaben, Abschlussarbeiten und extracurriculare Aktivitäten vorzustellen. Das Symposium bietet die Möglichkeit, vor einem akademischen Publikum Ideen, Pläne und Ergebnisse zu präsentieren und zu diskutieren. Darüber hinaus sind Doktoranden eingeladen, ihre Promotionsprojekte mit einem Poster zu präsentieren, um dadurch Feedback von anderen jungen Wissenschaftlern und Professoren für ihre wissenschaftliche Arbeit zu erhalten.

Ein Studierendensymposium bietet den Studierenden die Möglichkeit, akademische Konferenzen vor Ort kennenzulernen. Wie bei klassischen wissenschaftlichen Tagungen reichen sie Beiträge ein. Andere Studierende, Doktoranden, wissenschaftliche Mitarbeiter und Professoren bewerten und diskutieren in einem Gutachtenprozess die eingereichten Arbeiten. Interessante und gut ausgearbeitete Einreichungen werden zur Präsentation auf dem Symposium angenommen und die besten Arbeiten mit Preisen prämiert. Eine Studentenkonferenz unterscheidet sich damit kaum von einer anderen wissenschaftlichen Konferenz, wobei der Fokus des Gutachtenprozesses natürlich weniger auf einer rigiden Selektion liegt als vielmehr auf einer Unterstützung und Ermutigung der Einreichenden. Ein Ziel ist schließlich, das gegenseitige Verständnis von Lehrkräften und Studierenden zu fördern. Darüber hinaus ist das Studierendensymposium eine wichtige Plattform zu Vorstellung von und Kontaktaufnahme mit Unternehmen der Region.

Diese zweite Auflage des Studierendensymposiums Informatik an der TU Chemnitz hat 21 Einreichungen angezogen. Die Qualitätssicherung und Differenzierung wurde durch umfangreiche und detaillierte Hinweise zu Verbesserungen sowie eine Einordnung der Beiträge in lange Artikel, kurze Artikel mit entsprechenden Vorträgen sowie Poster und Demobeiträge erreicht.

Wir danken allen Autoren, Mitgliedern des Organisations- und Programmkomitees und der Chemnitzer Fachschaft Informatik für ihre Unterstützung und ihr Engagement bei der Ausrichtung der Veranstaltung. An dieser Stelle möchten wir uns ganz herzlich bei den Hauptsponsoren Intenta GmbH, DMK E-BUSINESS GmbH, 3DInsight GmbH sowie IT SONIX custom development GmbH für die Unterstützung bedanken.

Chemnitz,
April 2016

Max Eibl
Martin Gaedke

Organisations- und Programmkomitee

Maximilian Eibl	Co-Chair, Technische Universität Chemnitz
Martin Gaedke	Co-Chair, Technische Universität Chemnitz
Katharina Einert	Technische Universität Chemnitz
Janett Kühn	Fachschaftsrat Informatik, Technische Universität Chemnitz
Andreas Müller	Technische Universität Chemnitz

Frederik Beuth	Technische Universität Chemnitz
Marcus Hilbrich	Software Quality Lab (s-lab), Universität Paderborn
Danny Kowerko	Technische Universität Chemnitz
Jörg Lässig	University of Applied Sciences Zittau/Görlitz
Michael Martin	Universität Leipzig
Andreas Munke	Staatliche Studienakademie Glauchau
Michael Nagler	Technische Universität Chemnitz
Knut Odermann	Technische Universität Chemnitz
Philip Parsch	Technische Universität Chemnitz
Martin Reber	Technische Universität Chemnitz
Marc Ritter	Technische Universität Chemnitz
Paul Rosenthal	Technische Universität Chemnitz
Frank Seifert	Technische Universität Chemnitz
Peter Tröger	Technische Universität Chemnitz

Table of Contents

Full Paper

Holistische Beschreibung der Fähigkeiten heterogener Robotergruppen 1
Christine Jakobs

Automatisierte Verlässlichkeitsmodelle verteilter Anwendungen 13
Stefan Naumann

Nicht-blockierende Implementierung von verketteten Listen mit Hilfe der Intel Transactional Synchronization Extension 27
Thomas Jakobs

Konzeption und Implementierung einer Sprachsteuerung für Roboter 39
Kevin Kendzia

Evaluation von Touch-Gesten zur Steuerung eines nicht-humanoiden Roboters 51
Anna Zywietz

Entwurf und Implementierung von Optimierungs- und Funktionserweiterungen der hexagonalen Bildrasterung in der Videokompressionssoftware x264HMod 63
Tobias Schlosser, Robert Manthey and Marc Ritter

Entwicklung eines webbasierten Curve-Fitting Tools für komplexe Multiparameter-Funktionen 75
Titus Keller, Danny Kowerko and Marc Ritter

Entwicklung einer virtuellen Versuchsumgebung zur experimentellen Untersuchung von Raumorientierung und visueller Aufmerksamkeit 87
Sascha Jüngel, Frederik Beuth and Fred Hamker

Short Paper

Untersuchungen und Optimierung der Kontexterkennung auf mobilen Endgeräten mittels Sensorfusion 99
Maik Benndorf

Schallquellenlokalisierung zur Sprecheridentifizierung für Multi-User Dialogschnittstellen 105
René Schmidt and Wolfram Hardt

Reinforcement Learning with Object Localization in a Virtual Environment 111
Joseph Gussev, Helge Dinkelbach, Frederik Beuth and Fred Hamker

Automatische Berechnung des Ruiz-Kriteriums im Finite-Elemente-Postprocessing 117
Hannes Kuhnert

Learning of Spatial Invariances for Object-Ground Separation 127
Alex Schwarz, Frederik Beuth and Fred Hamker

Poster and Demos

Distributed Event-based Communication for Web Components 133
Michael Krug

COP: Improving Collaboration in e-Health Utilizing the Cloud 137
Fabian Wiedemann

CloudLinkSync: URI-basierte Adressierung lokaler Kopien von Cloud-Ressourcen 141
Frank Burian

Generisches Framework zur Parallelisierung von Echtzeitfähiger Bilddatenauswertung für Rekonfigurierbare Integrierte Schaltkreise 145
Stephan Blokzyl and Wolfram Hardt

Big Data Applications in Wireless Sensor Networks 149
René Bergelt and Wolfram Hardt

Incremental DOM for Web Components 153
Markus Ast

Optimierung der medizinischen Dokumentation durch den Einsatz assistierender spracherkennender Systeme - Konzept für ein Promotionsvorhaben 157
Christina Lohr

Attentive Robot Vision 161
Amirhossein Jamalian, Frederik Beuth and Fred Hamker

Author Index 165

Holistische Beschreibung der Fähigkeiten heterogener Robotergruppen

Christine Jakobs

TU Chemnitz, Fakultät für Informatik, Professur Betriebssysteme, Chemnitz
christine.jakobs@informatik.tu-chemnitz.de

Zusammenfassung. In der Robotertechnologie geht der Trend weg von speziell für einen Anwendungszweck entwickelten Plattformen hin zu universell einsetzbaren Robotern. Dies führt in der Praxis zu einem immensen Zuwachs an heterogenen Robotern. In dieser Arbeit beschreiben wir ein Modell zur Beschreibung der Fähigkeiten solcher heterogenen Roboter. Anwendungen, welche auf dieses Modell aufbauen, wird ermöglicht, die Heterogenität der Roboterplattformen gezielt auszunutzen. Dadurch wird es möglich, heterogene Gruppen von Robotern anhand ihrer Fähigkeiten zu kategorisieren und in dynamischen Gruppen autonom agieren zu lassen.

Schlüsselwörter: Roboter, Planung, Fähigkeiten, Datenmodell

1 Einleitung

Die Einsatzgebiete von Robotern, im Folgenden auch Roboterplattformen genannt, sind heutzutage weit gefasst. Sie werden in Produktionsstraßen der Industrie eingesetzt, um beispielsweise Autoteile zu montieren oder Bauteile zu lackieren. Ein neues Aufgabengebiet solcher Roboterplattformen sind *Smart Home*, *Internet of Things* oder *ubiquitous* Geräte. Hier werden verschiedene internetfähige Geräte verwendet, um die Menschen im Alltag zu unterstützen oder eine bessere Vernetzung der Geräte zu erreichen.

Zu Beginn der Forschung in den 1980er Jahren wurden Roboter rein singulär hinsichtlich ihrer Möglichkeiten betrachtet [17]. Aus diesem Forschungsgebiet entwickelte sich der Ansatz Gruppen von Robotern zu untersuchen. Anfangs wurden z.B. in der Schwarmintelligenz homogene Plattformen verwendet. Mit steigender Anzahl der Sensoren und Aktuatoren bei Robotern wurden auch die Aufgabengebiete, in welchen diese Plattformen eingesetzt werden konnten, immer komplexer und umfangreicher. Die Verwendung von multifunktionalen, homogenen Robotern, ist in Einsatzgebieten mit unterschiedlichen Teilaufgaben meist nicht sinnvoll und zu kostenintensiv. Vielmehr wird der Fokus auf spezialisierte Roboter für die einzelnen Aufgabenteile gelegt. In der Industrie werden so ganze Produktionsstraßen von verschiedenartigen Robotern betrieben. Hierfür ist eine entsprechende Koordination der einzelnen Roboterplattformen notwendig, um einen reibungslosen Ablauf zu garantieren. Daher rücken Gruppen heterogener Roboter mehr in den Fokus der Forschung. Als heterogene Gruppe von Robotern

werden hierbei Roboterplattformen bezeichnet, welche sowohl hinsichtlich Form und Größe, als auch Sensoren und Aktoren unterschiedlich konzipiert sind. Sie unterscheiden sich damit in ihren Möglichkeiten zum Wahrnehmen und Agieren mit ihrer Umwelt. Bei der Kooperation der Gruppenmitglieder nimmt die Komplexität der Steuerung des Gesamtsystems durch die Heterogenität der Roboter immens zu.

Sollen Aufgaben mit Hilfe einer heterogenen Gruppe erledigt werden, so stellt sich die Frage nach der Möglichkeit zur Automatisierung der Aufgabenplanung und -durchführung. Das von uns geplante Framework soll es einer heterogenen Gruppe von Robotern ermöglichen, gegebene Anforderungen in physikalische Handlungen umzuwandeln. Hierfür wird davon ausgegangen, dass eine dynamische Menge an verfügbaren Robotern vorliegt. Das bedeutet im Besonderen, dass sich diese Menge jederzeit durch Ausfall oder Aktivierung von Robotern verändern kann. Die zu erledigenden Aufgaben werden in Ketten von Teilaufgaben zerlegt und durch eine, den Anforderungen entsprechend geformte Gruppe von Robotern ausgeführt. Dabei sollen bei Bedarf unterschiedliche Möglichkeiten der Problemlösung gegeneinander abgewogen und bewertet werden. Bei der Ausführung soll unser System befähigt sein, automatisch auf Veränderungen, z.B. durch Umwelteinflüsse oder Ausfälle zu reagieren.

In einem ersten Schritt ist ein übergreifendes Datenmodell sinnvoll, welches eine Abstraktion der Roboter auf ihre Handlungsfähigkeit erlaubt. Dieses soll die Unterschiede zwischen den einzelnen Plattformen mit größtmöglicher Genauigkeit erfassen, dabei aber noch generell genug sein, um eine einfache Integration neuer Hardware zu ermöglichen. Die so entstehende Hardwareabstraktion beschreibt die funktionalen und technischen Roboteraspekte sowie deren Güte. Dadurch wird es ermöglicht aus einem Pool von verfügbaren Robotern eine dynamische Gruppe aufgrund der physikalischen Parameter und Grenzen der Roboter auszuwählen. Die Auswahl kann hierbei nicht nur hinsichtlich der Verfügbarkeit, sondern auch bezüglich der benötigten Güte von Sensoren und Aktuatoren erfolgen. Die ausgewählte Gruppe von Robotern soll befähigt sein, ihnen zugeteilte Aufgaben autonom, dynamisch und adaptiv auszuführen. Das Datenmodell sollte flexibel genug sein, um Hardware auch über Roboter hinaus beschreiben zu können. Dies würde die Voraussetzungen schaffen, um zukünftig auch *ubiquitous* Geräte integrieren zu können. So kann das System bei der Planung der Arbeitsschritte sowohl Sensorwerte der Roboter als auch Umgebungswerte bspw. von Thermometern oder Überwachungskameras verwenden, um eine möglichst optimale Lösung zu erarbeiten.

Wir geben in diesem Artikel einen Überblick über das Datenmodell und legen so die Grundlage für weitere Forschungsarbeiten. Hierfür wird zunächst ein Überblick über verwandte Arbeiten gegeben. Im Anschluss daran werden wir das Modell vorstellen und die Funktionalität anhand eines Beispiels beschreiben. Abschließend geben wir einen Überblick über den weiteren Verlauf der Forschungsarbeit.

2 Verwandte Arbeiten

Die verwandten Arbeiten können grob in zwei verschiedene Bereiche gegliedert werden. Zum einen existieren Ansätze, welche sich mit der Hardwareabstraktion auseinandersetzen. Zum anderen gibt es zahlreiche Projekte, in welchen vor allem die Problemstellung der Planung von Aufgaben im Vordergrund steht. Die Hardwareabstraktion heterogener Robotersysteme ist ein Gebiet der Forschung, in dem es noch keine generellen Datenschemata gibt, sondern vielmehr kleinere, für einzelne Spezialanwendungsgebiete geschaffene Ansätze existieren.

Lemaire [11] untersucht den Aufbau eines vierstufigen Abstraktionssystems, mit dessen Hilfe unterschiedlichste Arten von *ubiquitous* Geräten beschrieben werden können. Hierbei wird für jedes Gerät ein eigenes Modell erstellt, welches sowohl die Fähigkeiten, als auch die Zustände und Dienste der Geräte enthält. In unserem Ansatz wollen wir ein generell einsetzbares Datenmodell erstellen, in welchem die Fähigkeiten aller derzeit im System aktiven Roboter enthalten sind.

Das CoABS Projekt [10] beschäftigt sich mit dynamischen Mengen von Agentensystemen. Im Fokus steht hierbei weniger die Bildung von Gruppen von Ausführungseinheiten, als vielmehr der Versuch eine Klassifizierung großer Mengen von Geräten. Unser Ansatz dagegen hat nicht zum Ziel eine möglichst große Anzahl von Geräten zu verwalten, sondern eine stabile Planungsgrundlage zu schaffen in der Flexibilität und Erweiterbarkeit eine wichtige Rolle spielen.

Das Fehlen eines Modells zur Hardware-Abstraktion für Robotersysteme veranlasste die Object Management Group zur Veröffentlichung eines *Request for Proposal* [14]. Die Problembeschreibung und die Voraussetzungen, welche das Modell erfüllen soll, betrachten jedoch nur eine Abstraktion von Roboterbauteilen und deren Treiber in eine darüber liegende *Middleware*. Dadurch soll das Erstellen einer gemeinsamen Programmierschnittstelle für eine Vielzahl von Robotern erleichtert werden. Hierbei wird jedoch in den Mindestanforderungen die Bewertung der Güte sowie die strukturierte Darstellung der Komponenten nicht betrachtet. Diese Daten sind jedoch für unsere Hardware-Abstraktion, die auch Gruppen von heterogenen Systemen kooperativ Aufgaben erfüllen lassen soll, zur optimalen Anwendbarkeit von Planungsalgorithmen nötig.

Die *Distributed Management Task Force (DMTF)* hat den *Common Information Model (CIM)* [6] Standard verabschiedet. Mit diesem steht eine einheitliche, anbieter- und plattformunabhängige Managementschnittstelle zur Verfügung. Auch wenn es Bestrebungen gab, CIM auf Roboterplattformen auszudehnen, ist es derzeit noch nicht möglich, den Standard hierfür zu nutzen [4].

Das *Robot Operating System (ROS)* [15] stellt Bibliotheken und Hilfsprogramme zur Verfügung, um die Programmierung von Roboteranwendungen zu unterstützen. Das Open-Source Projekt stellt hierfür sowohl eine Hardware-Abstraktion zu Treibern der unterschiedlichen Roboterplattformen, als auch eine Nutzerschnittstelle zur Verfügung. Auch wenn in ROS mehrere Roboter simultan verwendet werden können, bietet es bisher keine Möglichkeit, Gruppen von Robotern durch ein gemeinsames Datenmodell zu verwalten. Unser Bestreben

ist es, unsere Forschungsergebnisse in ROS zu integrieren, um es hinsichtlich der Behandlung von Robotergruppen zu erweitern.

Alle von uns betrachteten Ansätze zur Planung haben einen dedizierten Anwendungsfall, wie z.B. Drenjanac & Tonic [8] auf landwirtschaftliche Zwecke. In diesem Projekt ist die Autonomie auf grundlegende Funktionen, wie beispielsweise Wegkorrekturen beschränkt. Unser System soll allgemeine Anwendungen abdecken und hierbei innerhalb der Fähigkeiten von Robotern ein vollständiges dynamisches und adaptives Planen sowie Ausführen ermöglichen.

Das Martha Projekt [2] beschäftigt sich mit der Problematik des Umladens an Häfen, Flughäfen und Rangierbahnhöfen. Alami et al. beschränken hierbei die Kooperation der Roboter auf Absprachen bzgl. Ressourcennutzung und Veränderungen der Umwelt wie z.B. Hindernissen. Die Aufgaben werden von einem zentralen Controller an die einzelnen Roboter verteilt. Diese kombinieren daraufhin ihre Pläne, um eine Ablaufkoordination in der Nutzung von Ressourcen zu erstellen. Unser Projekt soll jedoch eine gezielte Zusammenarbeit von Robotern in der Ausführung ihrer Aufgabe ermöglichen und fördern. Deshalb ist eine wesentlich weitreichendere Kooperation zwischen den Gruppenmitgliedern nötig. Zudem soll unser System es ermöglichen, Aufgaben gezielt an Gruppen von Robotern zu verteilen. Im Gegensatz zum Martha Projekt ist es unser Ziel, ein System zu erschaffen, in welchem eine Modellierung der Umwelt nicht notwendig ist. Deshalb ist unser Ausgangspunkt die Modellierung der vorhandenen Fähigkeiten von Robotern und die daraus resultierenden zur Verfügung stehenden Informationen über die Umwelt.

Von MacKenzie [12] wird der Ansatz verfolgt, die Teilaufgaben (Tasks) durch einen zentralen Kontrollpunkt über ein Auktionssystem zu verteilen. Die Kooperation zwischen den Gruppenmitgliedern beschränkt sich auf Absprachen beispielsweise bezüglich der Reihenfolge der Erledigung von Teilaufgaben sowie Kollisionsvermeidung. Ein gemeinsames, kooperatives Arbeiten der Gruppe ist hierbei nicht gegeben.

Parker [16] beschreibt eine Softwarearchitektur, welche eine fehlertolerante Kontrolle von Gruppen heterogener Roboter ermöglicht. Das ALLIANCE-Projekt setzt auf Mengen von festgelegtem Verhalten wodurch die Flexibilität des Einsatzes von Sensoren und Aktuatoren eingeschränkt wird. Eine flexible Erweiterung um neuartige Technologien ist dadurch nur durch tiefe Eingriffe in das System möglich. Unser Datenmodell ist dagegen möglichst flexibel und abstrakt gehalten, um derartige Eingriffe unnötig zu machen.

Bei der *PEIS-Ecology* aus [5] werden alle verfügbaren Geräte als Ökologie aufgefasst. Das auf Linda [1] aufbauende Projekt hatte zum Ziel, ein Lebensumfeld zu schaffen, in dem die Roboter selbstständig die Hausarbeit erledigen. Zu diesem Zweck sind die Haushaltsgegenstände als Ökologie eingebunden und das Umfeld untersteht einer ständigen Beobachtung durch Kameras. Das Modell baut darauf auf, dass die Geräte alle benötigten Informationen aus ihrem Umfeld *erfragen*. Durch dieses Konzept wird ein immens hohes Maß an Kommunikation benötigt. Unser Ansatz soll auf eine Kombination aus Umgebungswerten und

eigenen Sensorwerten und der anderer beteiligter Geräte aufbauen. Zudem soll eine umfassende Überwachung des Einsatzgebietes vermieden werden.

In den bisher genannten Arbeiten wird kein Überblick über das verwendete Datenmodell gegeben. Dies erschwert einen genaueren Vergleich von verschiedenen Ansätzen und deren Weiterentwicklung. Die Form der Darstellungen führt jedoch zu dem Schluss, dass die Modelle weniger auf das gezielte Herausarbeiten von Heterogenität ausgerichtet sind. Vielmehr handelt es sich meist um vordefinierte und bekannte Systeme, Umgebungen und Aufgaben.

3 Datenmodell

Das Datenmodell (*Capability Model*) soll die Möglichkeit bieten, aus einer gegebenen Kette von Aufgaben die für die Ausführung nötigen Sensoren und Aktuatoren auszuwählen. Aufgrund dieser Auswahl soll es möglich sein, eine ideale Konfiguration aus Robotern zusammenzustellen welche die Teilaufgaben autonom durchführt. Im Verlauf der Ausführung dieser Aufgaben wird regelmäßig eine Evaluation der gewonnenen Informationen stattfinden um gegebenenfalls gegensteuern zu können. Dieses Verhalten stellt, wie in Abbildung 1 gezeigt, einen geschlossenen Regelkreis dar.

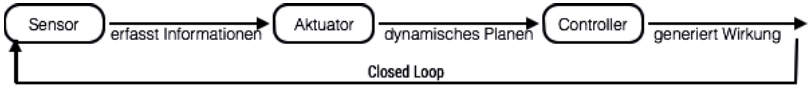


Abb. 1. Regelkreis im Roboterbetrieb

Im ersten Schritt werden zur Planung der Ausführung einfache Planungsalgorithmen verwendet. Diese gehen grundsätzlich davon aus, dass Ressourcen unbegrenzt zur Verfügung stehen. Deshalb lassen wir im ersten Schritt Beschränkungen, wie z.B. die Größe und der Bewegungsradius der Roboter außen vor. Durch die Wahl eines kleinen, frei zugänglichen Aktionsumfeldes, sowie kleinteilige Aufgaben stößt der Roboter nicht an energietechnische Grenzen. Deshalb werden auch Störungen aus der Umwelt nicht explizit betrachtet. Aus diesen Vorüberlegungen ergibt sich der folgende Aufbau des Datenmodells, sowie dessen Erweiterung um Metriken und Güteangaben.

3.1 Aufbau

Generell gliedert sich das Datenmodell in mehrere hierarchische Ebenen. Hierbei wird in Schicht 1 (Abbildung 2), abgeleitet aus dem Regelkreis, zwischen Sensoren und Aktuatoren unterschieden.Manche Bauteile fungieren gleichzeitig als Sensor und Aktuator, wie z.B. Kommunikationsmedien. Diese werden in beiden Bereichen eingruppiert, um eine vollständige Abbildung der Roboter zu schaffen.

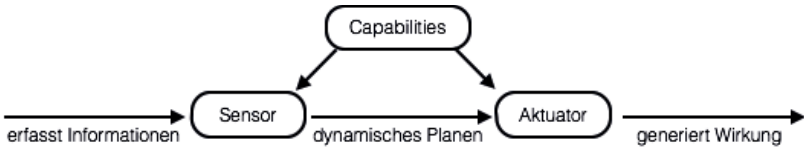


Abb. 2. Schicht 1: Gliederung des Capability Model in Sensoren und Aktuatoren

Sensoren sind in sich abgeschlossene Komponenten, welche nachgeordneten Einheiten erfassend oder messend Daten zur Verfügung stellen [9]. Diese Daten können vom System interpretiert und so als erfasste Information verwendet werden. In unserem Datenmodell erfolgt die Gruppierung der Sensoren derart, dass sowohl der Benutzer, als auch der Planungsalgorithmus optimal damit arbeiten können. Der Benutzer muss die Gruppierung dahingehend nachvollziehen können, dass er neue Plattformen oder Sensoren hinzufügen kann, aber auch die Entscheidungen des Algorithmus nachvollziehen kann. Eine Möglichkeit wäre demzufolge, die Sensoren anhand der klassischen menschlichen Sinne einzuteilen: Hören, Tasten, Sehen, Riechen und Schmecken. Dies ist für den Algorithmus zur Planung überflüssig, da dieser die Zwischeninformationen nicht sinnvoll verarbeiten kann.

Wir haben uns daher entschieden, die Sensoren hinsichtlich der verwendeten Technologie, z.B. Licht, Funk und Druck, einzusortieren. Die somit entstandene zweite Ebene für Sensoren (Abbildung 3) ist ebenfalls für den Menschen verständlich und leicht erweiterbar. Diese Zwischenebene kann vom Planungsalgorithmus mit verwendet werden, um durch bekannte Umweltbedingungen (bspw. fehlende Beleuchtung) bestimmte Technologien auszuschließen. Diese Aufteilung führt allerdings auch dazu, dass gegebenenfalls Sensoren wie bspw. Whisker (Zug/Druck) mehrfach eingruppiert werden. Außer Acht gelassen werden bei unserer Betrachtung interne Sensoren der Roboter, welche Informationen zur Bewertung des Roboterzustands liefern, wie beispielsweise ein Thermometer für überhitzende Bauteile. Diese sind für den Planungsalgorithmus nicht relevant, da sie keine Auskunft über Fähigkeiten des Roboters oder Wahrnehmungen aus dessen Umwelt geben.

Aktuatoren bezeichnen in der Steuerungs- und Regelungstechnik das signalwandlerbezogene Gegenstück zu Sensoren und bilden die Stellglieder in einem Regelkreis. Sie setzen Signale einer Regelung in (meist) mechanische Arbeit um (Bewegungsregelung).

Es wäre möglich, die Aktoren ebenfalls anhand der verwendeten Technologie zu klassifizieren, jedoch haben Roboter aufgrund ihrer Bauform beschränkte Möglichkeiten zur Interaktion. Eine Klassifizierung hinsichtlich der Handlungsmöglichkeiten ist vorteilhaft, da sie, durch die beschränkten Möglichkeiten übersichtlich und nachvollziehbar ist. Für die Planung von Handlungen sind im ers-

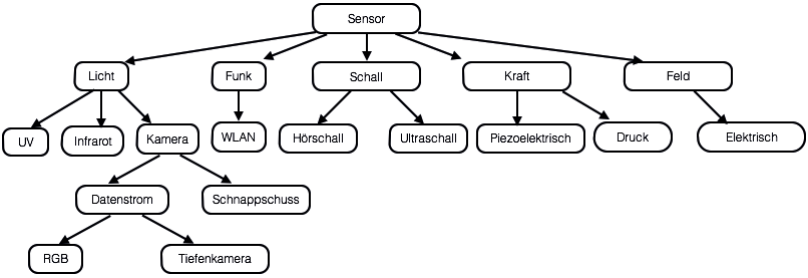


Abb. 3. Schicht 2a: Gliederung der Sensoren anhand unterschiedlicher Technologien (Auszug)

ten Anlauf die generellen Fähigkeiten, also die Möglichkeiten, die eine Wirkung zu erzielen, wichtiger. Die verwendete Technologie ist im zweiten Schritt wichtig, um anhand der gemessenen Umweltbedingungen bestimmte Aktorenformen auszuschließen. Deshalb werden die Aktuatoren vornehmlich hinsichtlich ihrer möglichen Verwendung aufgeteilt, danach anhand ihrer Technologie (siehe hierzu Abbildung 4). So können die Aktoren bspw. zur Fortbewegung des Roboters oder zur Manipulation von Gegenständen verwendet werden. Hierzu gibt es unterschiedliche Technologien wie bspw. Räder und Rotoren, Hände und Greifer, die jeweils eigene Vor- und Nachteile besitzen.

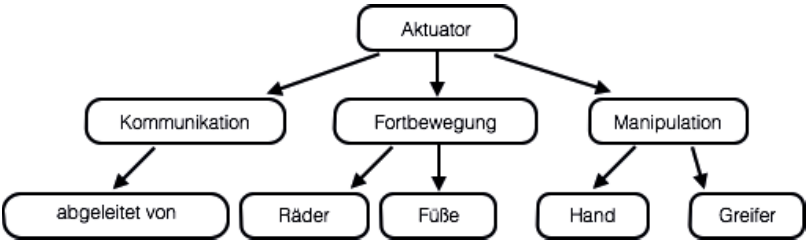


Abb. 4. Schicht 2b: Gliederung der Aktoren (Auszug)

Durch das bisher entstandene Modell ist es möglich, bestehende Roboter auf einfache Art und Weise zu kategorisieren. Anschaulich dargestellt, würden die Sensoren und Aktuatoren des humanoiden Nao Roboter von Aldebaran Robotics und des radbasierten X80 Pro von Dr. Robot wie in Abbildung 5 einsortiert werden. Die Gruppierungen in der zweiten Schicht, also die Technologien bei den Sensoren bzw. die Möglichkeiten zur Interaktion bei den Aktoren, werden im Folgenden, als Komponentenklassen bezeichnet.

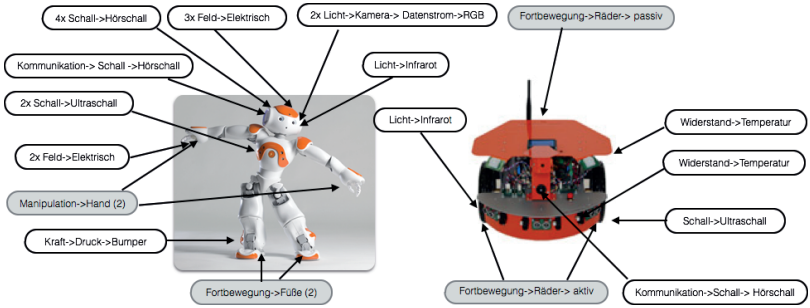


Abb. 5. Auszug aus der Kategorisierung zweier Roboter

3.2 Metriken und Güteangaben

Mit dem derzeitigen Aufbau des Modells ist eine Auswahl der Roboter basierend auf deren Sensoren und Aktuatoren möglich. Die reine Existenz eines Sensors bzw. Aktuators trifft jedoch keine Aussage darüber, ob dieser für die gestellte Aufgabe geeignet ist. Um eine optimale Grundlage für die Anwendung von automatischen Planungsalgorithmen [18] zu schaffen, werden deshalb sowohl in die Komponentenklassen als auch in die Einträge selbst Metriken und Güteangaben eingearbeitet. Die Metriken stellen hier die Frage, was in dieser Komponentenklasse als Bewertungskriterium ausschlaggebend ist. Durch die Güteangaben wird der Wert dieser Bewertung für das einzelne Bauteil angegeben. So kann bspw. die Auswahl einer Transportmöglichkeit beruhend auf der benötigten Kraft erfolgen.

Gemäß DIN EN ISO 80000-1 wird als Größe eine Eigenschaft bezeichnet, welche „einen Wert hat, der durch eine Zahl und eine Referenz angegeben werden kann“. Abgeleitete Größen sind dabei Größen, „die als Funktion der Basisgrößen“ definiert sind [13]. In unserem Datenmodell werden den Komponentenklassen Metriken in Form von Größen bzw. abgeleiteten Größen zur Einschätzung der Fähigkeiten zugewiesen. Die Sensoren und Aktuatoren verfügen über ein atomares Modell, welches die folgenden Daten enthält:

1. die ID des Devices und des zugehörigen Roboters,
2. den Ort,
3. die Komponentenklasse, zu welcher das Gerät gehört und
4. die Gütekriterien bezogen auf die Metrik der Vaterklasse

Es ist darauf zu achten, dass die Angabe von Gütekriterien in den korrespondierenden Einheiten gemäß DIN EN ISO 80000 erfolgt. Dies kann dazu führen, dass weitere Umrechnungen notwendig sind. Auf diese Weise kann sichergestellt werden, dass die Angaben bei der Planung richtig einfließen können. Dieser Teil des Modells ist noch nicht abschließend formuliert, wurde jedoch exemplarisch an realen Robotern evaluiert. Vor allem im Bereich der angegebenen Metriken

werden sich im Rahmen der Erweiterung des Modells weitere Veränderungen ergeben.

4 Beispiel

Zum Aufstellen unseres Modells wurden allgemeine Daten über verfügbare Sensoren und Aktuatoren verwendet. Für den anschließenden Test, ob das Modell allgemein genug und leicht erweiterbar ist, wurden verschiedene Roboter direkt klassifiziert. Im Rahmen der Beispiele in Tabelle 1 und 2 soll vor allem die Möglichkeit der Klassifizierung verdeutlicht werden. Deshalb wurden die technischen Angaben aus den Dokumentationen ohne weitere Umrechnungen verwendet. Zur besseren Lesbarkeit und Übersicht, wurde auf die Angabe von ID und Position der jeweiligen Komponente verzichtet. Sich in den Beispielen ergebende Lücken der Werte können mit wenig Aufwand im Rahmen der Implementation vervollständigt werden. In den Tabellen werden Sensoren mit „S“, Aktoren mit „A“, Ultraschall mit „US“, Datenstrom mit „D“ und Kommunikation mit „K“ abgekürzt. Die Richtungsanzeigen werden mit „L“ für links, „R“ für rechts, „M“ für mittig und „F“ für vorne abgekürzt.

4.1 Nao Roboter

Als erstes Beispiel (Tabelle 1) wurde der humanoide Roboter Nao von Aldebaran Robotics in der Version 4 klassifiziert [3]. Die Positionsangaben bei Robotern mit beweglichen Teilen, beispielsweise wie hier bei einem humanoiden Roboter, beziehen sich immer auf unterschiedliche Nullpunkte. Hier muss die tatsächliche Position der Sensoren oder Aktuatoren in einem weiteren Schritt mit Bezug zur aktuellen Haltung berechnet werden.

Tabelle 1: Klassifizierung Nao Roboter

Name	**Klasse**	**Güte**
3x Head/Touch	S→Feld→Elektrisch	
3x L-/R-Hand/Touch	S→Feld→Elektrisch	
2x L-/R-Foot/Bumper	S→Kraft→Druck→Bumper	
US/Left/S	S→Schall→US	40kHz; 1cm; 0.25-2.55m
US/Right/S	S→Schall→US	40kHz; 1cm; 0.25-2.55m
US/Left/Transmitter	A→K→Schall→US	40kHz; 1cm
US/Right/Transmitter	A→K→Schall→US	40kHz; 1cm
2x Speaker	A→K→Schall→Hörschall	
4x Microphone	S→Schall→Hörschall	300Hz - 8kHz
2x Camera	S→ Licht→Kamera→D→RGB	1280*960@30 fps
2x IR	S→Licht→IR	940nm
Gyro	S→Kraft→Piezoelek.→Gyro	500°/s
Acceleration	S→Kraft→Piezoelek.→Beschl	2g
WLAN	A→K→Funk→WLAN	IEEE 802.11 b/g/n
LLeg/RLeg	A→Fortbewegung→Füße(2)	
LArm/RArm	A→Manipulation→Hand(2)	

4.2 X80 Pro

Als Beispiel für einen fahrenden Roboter (Tabelle 2) wurde der X80 Pro von Dr. Robot verwendet [7]. Beim Roboter X80 Pro ergaben sich bei der Einarbeitung Schwierigkeiten, da die vorliegende Dokumentation manche Daten nicht enthält. Hier ist zu prüfen, ob diese im Rahmen des SDK auslesbar oder anderweitig verfügbar sind.

Tabelle 2: Klassifizierung X80 Pro

Name	**Klasse**	**Güte**
2x aktiv Wheel	A→Fortbewegung→Räder→aktiv	75cm/s
1x passive Wheel	A→Fortbewegung→Räder→passiv	
WLAN	A→K→Funk→WLAN	IEEE 802.11g
6x Ultrasonic	S→Schall→US	40kHz; 10-340cm
7x IR	S→Licht→Infrarot	
Temperature	S→Widerstand→Temperatur	-20/+80 °C
Camera	S→Licht→Kamera→D→RGB	352*288@15fps
2xHuman Detection	S→Licht→Infrarot	60-100°; 5-14 μm
Acceleration	S→Piezoeletr.→Beschleunigung	+/- 1g
Speaker	Aktor→K→Schall→Hörschall	320Hz - 20kHz
Microphone	S→Schall→Hörschall	20Hz - 20kHz

4.3 Planungsbeispiel

Würde man den vorhergehend klassifizierten Robotern die Aufgabe „Hole den Kasten Wasser und folge mir“ geben, so würde sich der Planungsablauf wie in Abbildung 6 ergeben. In diesem Beispiel wurde bereits ein Vorgriff auf die weitere Forschungsarbeit gemacht, indem bestimmte Softwarefähigkeiten, welche erst im Rahmen der weiteren Ausarbeitung des Projektes bewertet werden können, mit einbezogen wurden.

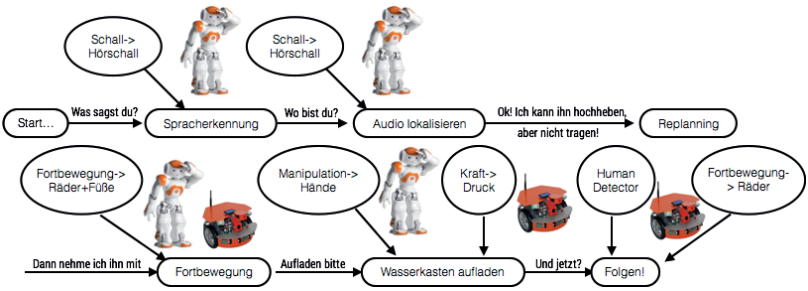


Abb. 6. Szenario - Hole den Kasten Wasser und folge mir

Zu Beginn würde der Nao mit seinem eingebauten Modul zur Spracherkennung (Sensor, Softwarefähigkeit) und räumlichen Hören (Audio lokalisieren, Sensor, Softwarefähigkeit) versuchen, die gegebene Aufgabe akustisch zu verstehen und den Ort des Auftragstellers zu erkennen. Der Planungsalgorithmus würde feststellen, dass der Kasten Wasser für ihn zum Tragen zu schwer ist (Güteangabe des Roboters). Das daraufhin ausgelöste neue Planen (Replanning) würde ergeben, dass sich der X80 Pro zusammen mit dem Nao zum Aufgabensteller hin begibt (Aktuatoren). Daraufhin würde der Nao den Kasten Wasser auf den X80 Pro aufladen (Aktuator). Dieser würde daraufhin mit seinem Human Detector (Sensor) befähigt sein, dem Auftragsteller zu folgen (Aktuator).

5 Ausblick

Das hier vorgestellte Datenmodell ermöglicht die Klassifizierung von Roboterplattformen derart, dass die Heterogenität ihrer Sensoren und Aktuatoren herausgearbeitet und so gezielt ausgenutzt werden kann. Im weiteren Verlauf soll zunächst eine Generalisierung der Metriken und Güteangaben erfolgen, um die dargestellte Nutzung zu ermöglichen. Als Gegenstück zur Abstraktion der Fähigkeiten von Robotern soll ein zweites Datenmodell für die Funktionalitäten der verwendeten Software zum Betrieb entstehen. Hierin werden die Möglichkeiten zum Verarbeiten der gewonnenen Sensorinformationen und die Möglichkeiten zur Steuerung der Roboter erfasst. So kann ein Vergleich zwischen beiden Modellen für die Planung verwendet werden. Dadurch ist es nicht mehr nötig, dass jede Funktionalität der Roboter immer verwendbar ist, da bspw. beim Ausfall einer Softwarekomponente der dazugehörige Sensor oder Aktuator zwar generell noch verfügbar ist, aber in der Planungsphase verworfen wird, da er nicht ausgewertet bzw. gesteuert werden kann.

In einem nächsten Schritt soll das erstellte Datenmodell implementiert und in ROS integriert werden. Im weiteren Verlauf können damit erste Planungsansätze umgesetzt und evaluiert werden. Mit den daraus gewonnenen Erkenntnissen kann eine Möglichkeit entwickelt werden, weitere Constraints aus Umwelt und Plattformen zu berücksichtigen. Damit ist die Grundlage für die dynamische Bildung von Gruppen zur Lösung von Aufgaben gelegt.

Literatur

1. Ahuja, S., Carriero, N., Gelernter, D.: Linda and friends. Computer 19(8), 26–34 (1986)
2. Alami, R., Fleury, S., Herrb, M., Ingrand, F., Robert, F.: Multi robot cooperation in the MARTHA project. IEEE Robotics Automation Magazine 5(1), 36–47 (1998)
3. Aldebaran Robotics: Nao documentation, http://doc.aldebaran.com/2-1/home_nao.html
4. Bernhardt, R., Dillman, R., Hormann, K., Tierney, K.: Integration of Robots into CIM. Springer Science & Business Media, Dordrecht (1991)
5. Broxvall, M.: A middleware for ecologies of robotic devices. In: Proceedings of the 1st International Conference on Robot Communication and Coordination. pp. 1–8. RoboComm '07, IEEE Press, USA (2007)
6. Distributed Management Task Force: Common information model, http://www.dmtf.org/standards/cim
7. Dr Robot: X80 wifi mobile robot development platform pro version, http://www.drrobot.com/products_item.asp?itemNumber=X80pro
8. Drenjanac, D., Tomic, S.D.K.: Middleware challenges in robotic fleets for precision agriculture. Journal of Mechanics Engineering and Automation 3(11), 702–714 (2013)
9. Heinrich, B., Linke, P., Glöckler, M.: Grundlagen Automatisierung Sensorik, Regelung, Steuerung. Springer Vieweg, Wiesbaden (2015)
10. Kahn, M.L., Cicalese, C.D.T.: Coabs grid scalability experiments. In: Second International Workshop on Infrastructure for Scalable Multi-Agent Systems at the Fifth International Conference on Autonomous Agents. ACM Press (2001)
11. Lemaire, O., Ohba, K., Hirai, S.: Dynamic integration of ubiquitous robotic systems through capability model processing. In: SICE-ICASE, 2006. International Joint Conference. pp. 1207–1211. Korea (2006)
12. Mackenzie, D.C.: Collaborative tasking of tightly constrained multi-robot missions. In: In Multi-Robot Systems: From Swarms to Intelligent Automata: Proceedings of the 2003 International Workshop on Multi-Robot Systems. vol. 2, pp. 39–50. Kluwer Academic Publishers (2003)
13. Normenausschuss Technische Grundlagen (NATG) im DIN: Größen und einheiten - teil 1: Allgemeines (iso 80000-1:2009 + cor 1:2011) (2013)
14. Object Management Group: Hardware abstraction layer for robotic technology (hal4rt) request for proposal. OMG Document: robotics/2014-06-14 (2014)
15. Open Source Robotics Foundation: Robot operating system, http://www.ros.org
16. Parker, L.E.: Alliance: an architecture for fault tolerant multirobot cooperation. IEEE Transactions on Robotics and Automation 14(2), 220–240 (1998)
17. Parker, L.E.: Springer Handbook of Robotics, chap. Multiple Mobile Robot Systems, pp. 921–941. Springer Berlin Heidelberg, Berlin, Heidelberg (2008)
18. Schuschel, H., Weske, M.: Automated planning in a service-oriented architecture. In: Enabling Technologies: Infrastructure for Collaborative Enterprises, 2004. WET ICE 2004. 13th IEEE International Workshops on. pp. 75–80 (2004)

Automatisierte Verlässlichkeitsmodelle verteilter Anwendungen

Stefan Naumann

Technische Universität Chemnitz, Professur Betriebssysteme
stefan.naumann@s2012.tu-chemnitz.de

Zusammenfassung. Verteilte Systeme werden immer größer und dynamischer. Knoten und Prozesse können zur Laufzeit ersetzt, hinzugefügt und entfernt werden. Die Verlässlichkeitsmodellierung dieser Systeme ist mit herkömmlichen oder manuellen Methoden sehr aufwändig. Hier wird gezeigt, dass es möglich ist aus aufgezeichneten Kommunikationsdaten Modelle verteilter Systeme zu erstellen und diese auf ihre Verlässlichkeit hin zu untersuchen. Dabei werden Graphen automatisiert erzeugt, die das verteilte System darstellen - erst mit einer groben Rechnersicht, später mit Prozessen als Knoten, bis hin zu gerichteten Kanten als Abhängigkeiten zwischen Prozessen. Diese Graphen werden einer Brückenerkennung unterzogen, um Single Points of Failure im System zu finden.

1 Einführung

Computersysteme werden vernetzter, Unternehmen, wie bspw. Soundcloud, setzen für ihre Produkte nicht mehr auf einzelne Großrechner, sondern auf viele (schwächere) Rechner, die nur durch Zusammenarbeit und Kommunikation über ein Netzwerk ihr Ziel erreichen können. Dabei bietet ein Rechner einen Dienst für das System an, der wiederum von den anderen Rechnern benutzt wird, bspw. bietet ein Rechner eine Datenbank an, die von anderen Rechnern im Netzwerk als Datenspeicher benutzt werden kann.

Durch diese Aufteilung eines größeren Produkts in einzelne Dienste wird eine Performanzsteigerung erhofft. Zudem sind einzelne Rechner, welche jeweils Teilaufgaben erfüllen, einfacher zu warten als ein großer Monolith, dessen Ausfall oder Wartung zu einem Gesamtausfall des Systems führen würde.

Die Abschätzung der Verlässlichkeit solcher Systeme wird sowohl für Wissenschaftler als auch für Ingenieure schnell komplex. Nicht nur weil mehrere (eventuell heterogene) Knoten/Rechner im System vorhanden sind, sondern auch, weil eine Dynamik dadurch entsteht, dass zur Laufzeit Knoten ersetzt, ausfallen oder neu hinzugefügt werden können.

Dieser Artikel beschäftigt sich damit von solchen verteilten Systemen automatisiert zur Laufzeit Modelle zu erstellen. Die darin enthaltenen Informationen über die Kommunikation im Netzwerk erlaubt es dieses auf Schwachstellen (bspw. Single Point of Failure) zu untersuchen.

2 Verwandte Arbeiten

Verteilte Systeme können durch Generierung von Abhängigkeiten aus den aktuell auf einem Rechner geöffneten TCP-Verbindungen modelliert werden. Dies benötigt eine synchrone Abfrage der Verbindungen auf allen Computern, was schwer umzusetzen ist.

Uhle [4] hat eine manuelle Modellierung eines größeren verteilten Systems vorgenommen und ist zu dem Schluss gekommen, dass diese Art sehr umständlich und wenig handhabbar ist. Außerdem zeigte Uhle, dass es möglich ist aus Verbindungsdaten einen Abhängigkeitsgraphen einer laufenden dynamischen Infrastruktur zu erzeugen. Dabei ist kein menschliches Eingreifen vonnöten. Uhle hat als Fallstudie ein verteiltes System mit Linux-Rechnern und einer Bazooka-Deployment-Infrastruktur. Für die Untersucht hat Uhle das Tool *netstat* verwendet um die geöffneten Verbindungen zwischen Prozessen zu beobachten.

Eine weitere Möglichkeit zur Modellierung verteilter Systeme wäre, die eintreffenden Nachrichten eindeutig zu markieren, und aus der Anfrage resultierende weitere Anfragen ebenfalls mit dieser ID zu markieren, wie es Fonseca et al [2] vorschlagen. Diese Markierung wird durch ein hinzugefügtes Headerfeld im Protokoll realisiert. Es muss von den Rechnern bzw. Prozessen vermerkt werden, wenn eine markierte Nachricht sie erreicht. Damit ist es möglich den Weg, den eine Anfrage gegangen ist im Netzwerk, nachzuvollziehen. Es ist erkennbar, welche Dienste welche anderen Dienste im Netzwerk verwenden, d.h. von welchen Diensten sie abhängen. Allerdings ist es nicht in jedem Protokoll möglich, weitere Headerfelder hinzuzufügen. Deshalb wird auf die Veränderung von Anfragen verzichtet.

Buzacott [1] stellt einen Algorithmus vor, wie ein gewichteter Graph danach ausgewertet werden kann, ob ein Knoten zu einer bestimmten Zeit unerreichbar ist, weil alle Wege zum dem Knoten versagen. Die Kanten seien dabei mit Werten versehen, die angeben, wie wahrscheinlich es ist, dass eine Kante zu einem bestimmten Moment nicht verfügbar ist. Es lässt sich durch den im Artikel gegebenen Algorithmus berechnen, wie wahrscheinlich ein komplettes Abschneiden von Knoten im Graphen ist. Die Ausfallwahrscheinlichkeiten der Kanten könnten mit dem hier dargestellten Verfahren ermittelt werden, allerdings ist das recht unelegant, da die Verwaltung von zwei Zählvariablen auf beiden Seiten einer Kante (ein Sende- und ein Empfangsereigniszähler pro Seite) genügen würde um die Verfügbarkeit der Kanten zu ermitteln. Hier wird versucht Kommunikation zwischen Rechnern (bzw. Prozessen) in einen Kontext zu setzen, anstelle Verfügbarkeiten von Kanten zu ermitteln.

3 Aufzeichnung der Kommunikation

Mithilfe des Wissens über die Kommunikation zwischen den Rechnern im verteilten System kann erkannt werden, welche Rechner welche Dienste nutzen. Hierzu wird davon ausgegangen werden, dass eine Kommunikation zwischen Rechnern das Benutzen eines Dienstes darstellt.

Um diese Kommunikation abzufangen, wurde ein *Sniffer* implementiert. Dieses Programm wird auf jedem Rechner gestartet um, unter Zuhilfenahme der

Bibliothek PCAP, empfangene und gesendete Netzwerkpakete aufzuzeichnen und zu einer zentralen Datenbank zu senden. PCAP (Packet Capture Library) ist eine Bibiliothek, die es einem Programm ermöglicht Netzwerknachrichten, die gesendet oder empfangen werden, mitzuhören. Dabei werden folgende Informationen pro Paket aufgezeichnet:

- HostId (eindeutige Nummer des aufzeichnenden Rechners)
- IP-Adresse des Empfängers
- IP-Adresse des Senders
- Zielport
- Quellport
- Zeitstempel (mit Nanosekunden)
- Anwendungsname (wenn verfügbar)

Jedem Rechner wird eine eindeutige HostId zugeordnet. In einer seperaten Datenbankrelation werden folgende Daten gespeichert, um zu wissen, welche Rechner zum Netzwerk gehören und welche Anfragen von außerhalb an das System gestellt werden:

- HostId
- Hostname
- eine Liste von:
  - IP-Adresse
  - Devicename (bspw. eth0)

Für die Auswertung der in der Datenbank gesammelten Daten wurde ein *Compute*-Programm implementiert, welches die Daten ausliest und mithilfe der unten dargestellten Verfahren ein Modell des Systems (Graph) erzeugt und dieses verarbeitet. Das *Compute*-Programm kann unabhängig von den *Sniffer*n ausgeführt werden, sodass eine spätere Auswertung der Daten möglich ist, auch wenn die *Sniffer* zu dem Zeitpunkt bereits abgeschaltet worden sind.

4 Modelle

Jedes der dargestellten Modelle ist in sich eine Momentaufnahme und nimmt ein statisches System an. Während der Aufzeichnung der Kommunikation sollten keine Änderungen im System passieren, d.h. Prozesse sollten nicht gestartet oder beendet werden und es sollten keine Rechner hinzugefügt oder entfernt werden. Die erstellten Modelle können drei verschiedene Detailgrade enthalten:

Die erstellten Modelle werden dabei durch Graphen dargestellt, es wurden drei Detailgrade der Modellierung betrachtet.

- Topologie mit Rechnern als Knoten, Kommunikation als ungerichtete Kante (Topologiegraphen)
- Prozesse als Knoten, Kommunikation als ungerichtete Kante (Prozessgraphen)
- Prozesse als Knoten, Dienstnutzung als gerichtete Kante (Abhängigkeitsgraphen)

Dabei werden alle Prozesse bzw. Rechner, die zum untersuchten verteilten System gehören als Knoten modelliert. Ein zusätzlicher Knoten stellt einen virtuellen Nutzer dar, der die Anfragen an das System sendet.

4.1 Topologiegraphen

In den Topologiegraphen wird Kommunikation zwischen Rechnern als ungerichtete Kante eingetragen. Wird ein Paket gefunden, werden die IP-Adressen zu HostIds aufgelöst, d.h. die beteiligten Knoten im Graphen werden identifiziert. Zwischen den beiden Knoten wird eine ungerichtete Kante eingefügt. Es entsteht genau ein Graph bei der Auswertung eines Netzwerktrace eines verteilten Systems.

4.2 Prozessgraphen

Bei der Erstellung der Prozessgraphen besteht die Schwierigkeit darin, die Prozesse als solche zu Erkennen. Ein Prozess wird auf einem Rechner ausgeführt und hat eine bestimmte Portnummer, angenommen diese ändert sich nicht. So ist es möglich Prozesse zu erkennen, d.h. durch den Rechnernamen und den Port eindeutig zu bezeichnen.

Das Verfahren basiert auf der Annahme, dass ein Empfangsereignis zu Sendeereignissen am gleichen Prozess führt. Um abschätzen zu können, welche folgenden gesendeten Nachrichten von dem Prozess gesendet wurden, der eine Nachricht erhalten hat, wird innerhalb eines Zeitintervalls ab Erhalt einer Nachricht nach gesendeten Nachrichten gesucht. Die beliebig konstante Zeitkonstante μ (mit $\mu > 0$) gibt dabei die Länge des Intervalls an.

Zur Erzeugung des Modells wird eine nach Zeitstempel sortierte Liste der Trace-Paketdaten durchsucht. Wird eine Nachricht gefunden, die noch nicht betrachtet wurde, wird eine ungerichtete Kante eingefügt. Am Zielknoten der Kommunikation wird nach weiteren Nachrichten gesucht, die innerhalb eines Zeitintervalls $[t, t + \mu]$ (t ist der zur Nachricht zugeordnete Timestamp, siehe Abschnitt 3) an andere Prozesse versendet wurden.

Es entsteht ein Graph pro Durchlauf mit konstantem Zeitwert μ.

4.3 Abhängigkeitsgraphen

Nutzt ein Prozess die Dienste eines anderen, wird dies als Abhängigkeit bezeichnet (d.h. der eine Prozess ist vom anderen Prozess abhängig). Die Nutzung von Diensten sei nur über Netzwerkkommunikation möglich, auch dann, wenn beide Prozesse auf der gleichen Maschine ausgeführt werden. Außerdem wird angenommen, dass jede Anfrage beantwortet wird. Jede weitere Anfrage vom Serverprozess ist notwendig um die erhaltene zu beantworten, d.h. es besteht ein kausaler Zusammenhang zwischen der erhaltenen Anfrage und kurz darauf versendeten Anfragen von einem Prozess.

Für die Darstellung dieses Modells wird ein Graph verwendet, wobei die Knoten Prozesse und die gerichteten Kanten eine Abhängigkeit ($A \to B \Rightarrow A$

ist abhängig von B) symbolisieren. Das Verfahren beginnt bei einer ins System eintreffenden Nachricht (einer Anfrage vom Nutzer) und geht rekursiv Wege durch das System ab. Wird eine Nachricht an einen Prozess erkannt, wird in der Liste von Nachrichten nach weiteren Nachrichten gesucht, die vom aktuellen Prozess aus innerhalb eines Zeitintervalls μ gesendet werden. Für jede Anfrage von außerhalb wird so ein neuer Graph erzeugt.

Abhängigkeitsgraphen ermöglichen das Erkennen von Pfaden im Netzwerk, die eine Anfrage wahrscheinlich genommen hat. Dadurch kann abgeschätzt werden, wie sich Ausfälle von Prozessen oder Verbindungskanten tatsächlich auf genau diese Anfrage auswirken würden.

Der Algorithmus könnte noch verbessert werden, indem nur solange bei einem Knoten nach folgenden Nachrichten gesucht wird, bis eine nächste Anfrage diesen Rechner erreicht. So könnten für größere μ die Anzahl der Kanten, die nicht kausal von der Anfrage abhängig sind, im Graphen verringert werden.

Im Algorithmus findet keine Prüfung statt, ob eine folgende Nachricht auch wirklich vom gleichen Prozess versendet wird. Dazu wäre es notwendig auf dem Rechner weitere Untersuchungen anzustellen und Prozesse genauer zu beobachten, um herauszufinden, ob tatsächlich dieser Prozess die Anfrage versendet hat und auch ob dieser Prozess das auch ohne das Empfangsereignis getan hätte. Das wiederum könnte schnell zu einer sehr großen und sehr unübersichtlichen Datenansammlung führen.

5 Verarbeitung der Modelle

5.1 Brücken und Artikulationspunkte

In [3] wird ein Algorithmus zur Erkennung von Brücken und Artikulationspunkten in zusammenhängenden einfachen Graphen beschrieben, der auf die oben dargestellten Modelle anwendbar ist.

Eine Brücke in einem Graphen ist eine Kante, deren Entfernen den Graphen in wenigstens zwei Teilgraphen zerfallen lässt; ein Artikulationspunkt ist ein Knoten, dessen Entfernen den Graphen in wenigstens zwei Teilgraphen zerfallen lässt. Der minimale Grad $\delta(G)$ gibt an, wie viele Ausgangskanten ein Knoten im Graphen mindestens aufweist.

Im vorgestellten Algorithmus wird eine Tiefensuche über dem ungerichteten Eingabegraphen durchgeführt. Alle Baumkanten werden zu den Blättern hin gerichtet, alle Kanten, die im Graphen aber nicht im Tiefensuchbaum enthalten sind, werden zur Wurzel hin gerichtet eingefügt. Allen Knoten wird eine von der Wurzel zu den Blättern aufsteigende Nummer zugeordnet und werden in dieser Reihenfolge (aufsteigend) besucht. Vom aktuellen Knoten werden nacheinander alle Rückkanten (also zu den Blättern führenden Kanten) verfolgt, und schließlich ein Weg zurück zur Wurzel gesucht. Die besuchten Kanten und Knoten werden markiert.

Die so gefundenen Wege werden Ketten (Chain) C_i genannt. Dieses Verfahren berechnet die Menge aller Ketten $C = C_1 + C_2 + \cdots + C_n$.

Es können anhand der Menge der Ketten folgende Eigenschaften für Knoten und Kanten abgelesen werden:

- G sei einfach. G heißt 2-kantenzusammenhängend genau dann wenn C die Kanten von G partitioniert, also restlos in Ketten einteilt.
- G sei 2-kantenzusammenhängend. G heißt 2-knotenzusammenhängend genau dann wenn C_1 der einzige Kreis in C ist.
- Eine Kante ist eine Brücke genau dann wenn sie in keiner Kette von C vorkommt.
- G sei einfach und der minimale Grad im Graphen sei $\delta(G) \geq 2$.[1] Ein Knoten v heißt Artikulationspunkt genau dann wenn v inzident zu einer Brücke ist oder v der erste Knoten eines Kreises in $C \setminus C_1$ ist.

Im Topologiegraphen stellt eine Brücke und ein Artikulationspunkt eine Single-Point-of-Failure dar. Fällt diese Kante bzw. der Knoten aus, zerfällt der Graph in mehrere Teilgraphen, d.h. im verteilten System ist eine Kommunikation zwischen zwei Teilen des Systems nicht mehr möglich.

Der vorgestellte Algorithmus arbeitet nur auf zusammenhängenden Graphen, allerdings ist es möglich, dass das Netzwerk nicht zusammenhängt. Ein Graph kann also aus wenigstens einer Zusammenhangskomponente bestehen. Der Algorithmus kann auf jeder der Komponenten einzeln ausgeführt werden. Dadurch werden Brücken und Artikulationspunkte unabhängig davon zurückgeliefert, ob der Graph zusammenhängt oder nicht.

6 Versuchsdurchführung

Zur empirischen Untersuchung des Ansatzes und der Auswertungsstrategien wurden die implementierten Anwendungen nach dem initialen Testen mit mehreren virtuellen Maschinen anhand eines Fallbeispiels auf echter Hardware untersucht.

6.1 Fallbeispiel

Abbildung 1 zeigt den logischen Aufbau des Testnetzwerks. Das Netzwerk besteht aus 14 Prozessen und dient als einfacher Formelrechner. Der Knoten ,,outside" (dt. außerhalb) dient als Verdeutlichung der Nutzerrechner/-prozesse. Dieser Knoten symbolisiert keinen Rechner/Prozess im Netzwerk, sondern stellt in einem Knoten alle anfragenden Rechner/Prozesse dar.

Die zu berechnende Formel besteht aus aneinandergereihten Operationen (Addition, Multiplikation, Potenzrechnung). Die Formel wird als String übermittelt.

Der Loadbalancer überprüft ob die Anfrage bereits gestellt wurde und liefert das gespeicherte Ergebnis aus der MySQL-Datenbank zurück. Existiert kein gespeichertes Ergebnis verteilt er die Anfrage an einen der Distributoren mithilfe des Round-Robin-Prinzips. Eine Fehlertoleranz ist nicht implementiert

[1] Diese Bedingung wird praktisch aufgelöst, da ein Beachten des Grades eines Knoten nicht implementiert wurde.

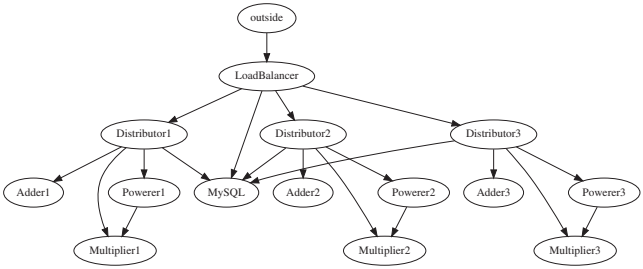


Abb. 1. Aufbau des Testnetzwerks - ein Formelrechner; Knoten symboliseren Prozesse

worden. Der Distributor interpretiert die Formel und verteilt die einzelnen Aufgaben an seine Prozesse, sodass der Adder alle Additionen, der Powerer die Potenzierungen und der Multiplier die Multiplikationen (einschließlich die des Powerers) übernimmt. Das Endergebnis wird vom Distributor in die Datenbank gespeichert. Es gibt drei Distributoren mitsamt der rechnenden Prozesse im System, einen Loadbalancer und eine MySQL-Datenbank.

Die Prozesse unterhalb eines Distributors werden nur von dem einen Distributor verwendet, d.h. Adder1 nur von Distributor1, Adder2 nur von Distributor2, etc.

6.2 Versuchsaufbau

In Abbildung 1 ist die logische Topologie des Testnetzwerks dargestellt. Dabei sind Prozesse zu sehen, die auf Rechner aufgeteilt werden müssen. Die Rechner sind allerdings sternförmig an einen Switch angeschlossen und in einem VLAN zusammengefasst. Der Switch kann alle Nachrichten dieses VLANs an einen beliebigen Port spiegeln. Allein auf diesem PC läuft die *Sniffer*-Anwendung, die die Netzwerkkarte im promiscuous mode betreibt um alle Nachrichten des VLANs mithören zu können. Auf diesem PC wird auch die Datenbank betrieben, in der die Nachrichten gespeichert werden. Ebenfalls dargestellt ist der Knoten ,,outside", der die Anfragen an das Netzwerk stellt.

Die implementierte *Sniffer*-Anwendung war nicht für einen solchen Einsatz gedacht, daher müssen vorab die Informationen über die Rechner (IP-Adressen, Namen, etc.) manuell gesammelt und in die Datenbank eingetragen werden. Zudem ist eine Abfrage des Prozessnamens, der eine Nachricht versendet hat, nicht möglich.

Bei der Aufteilung der Prozesse auf Rechner muss außerdem besonderer Wert darauf gelegt werden, dass Nachrichten nicht lokal ausgeliefert werden können, weil diese sonst den Switch und damit den *Sniffer* nicht erreichen. D.h. alle Prozesse, die in Abbildung 1 direkte Nachbarn sind, dürfen nicht auf demselben Rechner ausgeführt werden.

Beim eigentlich angedachten Vorgehen, einen *Sniffer*-Prozess pro Rechner zu starten, wäre es möglich gewesen die Datenbankanwendung mehrfach zu instantieren, d.h. bspw. pro Rechner einmal, sodass jeder *Sniffer* auf seinen eigenen Datenbankserver zugreift (*localhost*) um einen eventuellen Flaschenhalseffekt an der Datenbank zu verringern. Das ist hier nicht ohne Weiteres möglich, es müsste der *Sniffer* so implementiert werden, dass er nebenläufig nach Round-Robin-Prinzip auf mehreren Datenbankservern jeweils einen Teil der Nachrichten speichert.

6.3 Durchführung

Der Versuch besteht aus neun Rechnern, die das im Fallbeispiel definierte verteilte System ausführen. Es gibt einen zusätzlichen Rechner, der automatisiert Anfragen stellt. Das System verwendet das TCP-Protokoll.

Alle Anwendungen des verteilten Systems werden gestartet, d.h. die Verbindungen zum MySQL-Server bestehen zum Start der Aufzeichnung bereits. Nun wird die Aufzeichnung begonnen und das Skript auf dem „outside"-Rechner gestartet. Dieses stellt im Abstand von 10 Millisekunden vordefinierte Anfragen.

TCP hat die Eigenschaft, dass beim Verbindungsaufbau zu einem Port für die weitere Kommunikation der Port gewechselt wird. Das führt zu Problemen Kommunikation akkurat abzubilden. Daher wird der Versuch zweimal durchgeführt. Dabei werden:

- im 1. Durchlauf alle Nachrichten mit gesetztem TCP-PUSH-Flag aufgezeichnet
- im 2. Durchlauf alle Nachrichten mit ausschließlich gesetztem TCP-SYN-Flag aufgezeichnet.

Im ersten Durchlauf werden also alle Payload-Pakete aufgezeichnet[2], im zweiten nur die Pakete, die eine TCP-Verbindung initialisieren. Es wird davon ausgegangen, dass ein Prozess, der eine Verbindung aufbaut auch später eine Anfrage stellen will.

Testfälle Das Testskript sendet automatisch Anfragen an das System mit einem Abstand von 10 Millisekunden. Dabei wird kein Wert darauf gelegt bestimmte Anfragen zu versenden, es geht nur um die Menge. Es werden Formeln der Form $a\hat{}b * c + d$ an den Loadbalancer gesendet. Dieser wählt einen Distributor für die Weiterverarbeitung aus. Bei jeder Anfrage werden vom Distributor alle Rechenprozesse angefragt, auch dann, wenn die Variablen a, b, c oder $d = 0$ sind.

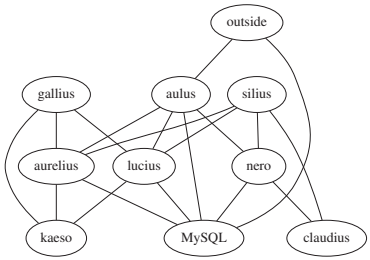
7 Ergebnisse

Es folgt die Auswertung der automatischen Modellierung und der Brücken- und Artikulationspunkterkennung anhand der beiden durchgeführten Versuchsdurchläufe.

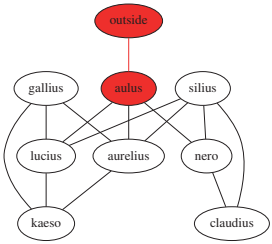
[2] davon ausgehend, dass alle Payload-Nachrichten mit gesetztem PUSH-Flag versendet werden

Rote Kanten sind erkannte Brücken, rot ausgefüllte Knoten stellen Artikulationspunkte dar. Die Graphen wurden mithilfe des GraphViz-Tools visualisiert.

7.1 Topologiegraph



(a) Topologiegraph erstellt mit PSH-Nachrichten



(b) Topologiegraph erstellt mit SYN-Nachrichten

Abb. 2. Topologiegraphen

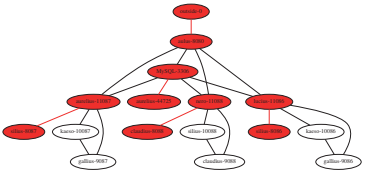
Abbildung 2 zeigt beide resultierende Topologiegraphen. Wie erwartet enthält der SYN-Graph den MySQL-Knoten nicht. Im PSH-Graphen ist eine Kante zwischen ,,outside" und dem MySQL-Knoten enthalten. Das könnte Kommunikation zwischen dem Switch oder einem nicht im System befindlichen Rechner und dem Rechner mit dem MySQL-Server sein. Eine Verbindung zu MySQL wird im Testskript nicht aufgebaut.

Beide Graphen stellen die verwendeten Kanten zwischen den Rechnern dar. Die Brücken- und Artikulationspunkterkennung bringt keine sinnvollen Ergebnisse hervor. Nur im SYN-Graphen werden überhaupt Brücken erkannt, nämlich die zwischen ,,outside" und dem Loadbalancer. Durch die dichte Vermaschung kann der Algorithmus keine Brücken bzw. Artikulationspunkte im Netzwerk erkennen.

7.2 Prozessgraphen

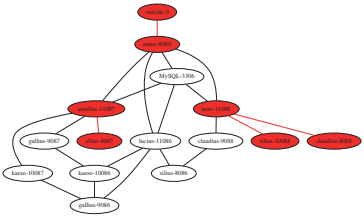
Abbildung 3 zeigt Prozessgraphen mit unterschiedlichen Zeitkonstanten μ. Das Verfahren wurde mit den Zeitkonstanten 1 ms, 5 ms, 10 ms und 50 ms, 500 ms, 5000 ms und 50000 ms durchgeführt. Die besten Ergebnisse lieferte der Threshold von 10 ms, was exakt der Zeit entspricht, die das Testskript zwischen zwei Anfragen wartet. In der PSH-Variante mit 10 ms Threshold erscheint eine Kante vom ,,MySQL" Knoten zum Rechner ,,aurelius" auf Port 44725. Das weißt darauf hin, dass hier eine Anfrage eventuell recht lang gedauert hat, somit dessen Sendeereignis unabhängig vom früheren Empfangsereignis gesehen wurde.

Graphen mit zu kleinem Threshold μ enthalten sehr viele Knoten und Kanten. Es zeigt sich, dass der Algorithmus Empfangs- und Sendeereignis nicht zusammenfassen kann, weil diese weiter als μ voneinander entfernt liegen. Hohe Thresholds führen zu falsch erkannten Kanten. Hier wird das nächste Sendeereignis noch zum früheren Empfangsereignis zugeordnet, obwohl eigentlich kein kausaler Zusammenhang besteht.

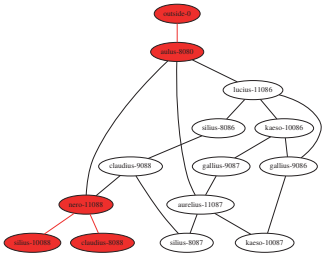


(a) Prozessgraph PSH-Nachrichten, $\mu =$ 10000 Mikrosekunden (10 ms)

(b) Prozessgraph SYN-Nachrichten, $\mu = 10000$ Mikrosekunden (10 ms)



(c) Prozessgraph PSH-Nachrichten, $\mu =$ 50000 Mikrosekunden (50 ms)



(d) Prozessgraph SYN-Nachrichten, $\mu = 50000$ Mikrosekunden (50 ms)

Abb. 3. Prozessgraphen mit unterschiedlichen Thresholds μ

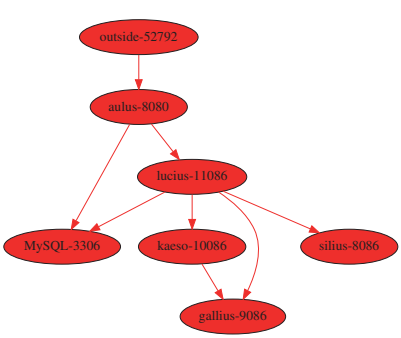
In realen Systemen mit variablen Verarbeitungszeiten oder wenn mehrere Anfragen nebenläufig im System bearbeitet werden ist es eventuell sehr schwer den Threshold passend zu wählen.

7.3 Abhängigkeitsgraphen

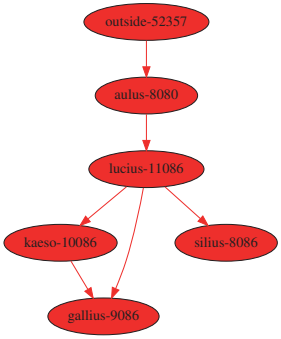
Mit jeder Anfrage an das System entsteht ein Graph, weshalb hier nur einige ausgewählte gezeigt werden können. Da die Kanten gerichtet sind, ist eine sinnvolle Brückenerkennung nicht mehr möglich, d.h. alle Kanten werden als Brücken und alle Knoten als Artikulationspunkte erkannt. Es gibt keine Abhängigkeiten im Netzwerk zu höheren Prozessen (d.h. keine Kreise). Das Verfahren erkennt auch keine Redundanzen.

1000 Mikrosekunden Es ist beim Threshold von 1 ms zu erkennen, dass der PSH-Graph eine Kante von ,,aulus" zu ,,MySQL" beinhaltet. Es kann also angenommen werden, dass in 1 ms der Loadbalancer eine Anfrage an den MySQL-Datenbankserver gestellt hat, aber im selben Zeitintervall keine Antwort bekommen hat bzw. keine weitere Anfrage an einen Distributor stellen konnte. Dieser Threshold ist zu klein um sinnvolle Aussagen über das System anstellen zu können.

5000 Mikrosekunden Mit 5 ms Threshold zeigt sich eine starke Varianz in den Graphen. Einige Graphen zeigen alle Kanten, die von der Anfrage genommen

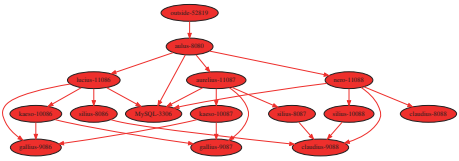


(a) Abhängigkeitsgraph mit PSH-Nachrichten

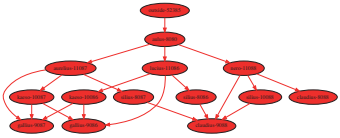


(b) Abhängigkeitsgraph mit SYN-Nachrichten

Abb. 4. Abhängigkeitsgraphen mit Threshold $\mu = 10000$ Mikrosekunden (10 ms)



(a) Abhängigkeitsgraph mit PSH-Nachrichten



(b) Abhängigkeitsgraph mit SYN-Nachrichten

Abb. 5. Abhängigkeitsgraphen mit Threshold $\mu = 50000$ Mikrosekunden (50 ms)

wurden. Andere stellen nur die Verwendung des Powerers und des Multipliers, wieder Andere den Weg bis zum Distributor. Fünf Millisekunden reichen somit nicht aus um dieses System akkurat zu erfassen.

Der PSH-Graph stellt den Weg einer Anfrage dar, allerdings fehlt bei den meisten Graphen die Kante vom Power-Prozess (Port 1008X) zum Multiplier-Prozess (Port 908X). Im Algorithmus wird bei jedem neuen Knoten nach Nachrichten gesucht, die 5000 Mikrosekunden nach Erhalt einer Nachricht versendet worden. Daher erschließt sich nicht ganz, warum gerade diese Kante fehlt. Der Distributor wartet bis eine Anfrage beantwortet wurde, und sendet erst dann eine neue, sodass es durchaus passieren kann, dass nicht alle drei Anfragen in dieses Zeitintervall fallen, allerdings passiert das beim Power-Prozess nicht.

10000 Mikrosekunden Abbildung 4 zeigt mit einem Threshold von 10000 Mikrosekunden (10 ms) erstellte Abhängigkeitsgraphen. Sowohl die PSH- als auch die SYN-Graphen sind sehr akkurate Darstellungen des Systems. In der PSH-Variante gibt es einige wenige Ausnahmen, die die Einfüge-Operation an den MySQL-Server vom Distributor nicht verzeichnet haben. Die Abhängigkeiten

zwischen den Prozessen sind dargestellt, auch vom Loadbalancer und Distributor vom MySQL-Server (in der PSH-Variante).

Prozessen wird hier genügend Zeit eingeräumt Anfragen zu bearbeiten und weitere zu stellen. Außerdem fällt ins Gewicht, dass Anfragen nie nebenläufig im System sind. In einem echten System kann diese Annahme nicht getroffen werden.

50000 Mikrosekunden Abbildung 5 zeigt zwei mit $\mu = 50$ ms erstellte Abhängigkeitsgraphen. Nachrichten von späteren Anfragen werden mit verarbeitet, sodass ein vollständigeres Bild vom System entsteht. Allerdings sind in beiden Graphen Kanten enthalten, die keine kausale Abhängigkeit zwischen Prozessen darstellen, sondern fehlerhafterweise früheren Empfangsereignissen zugeordnet wurden.

7.4 Fehlerbetrachtung

Beim Versuchsaufbau sind Fehler nicht komplett auszuschließen. Zum Beispiel wäre es möglich, dass einzelne Nachrichten aufgrund eines vollen PCAP-Buffers nicht aufgezeichnet wurden. Unter der Annahme, dass dass gleiche oder ähnliche Anfragen mehrmals an das System gesendet werden (eventuell auch von unterschiedlichen Nutzern), dann ist die Wahrscheinlichkeit, dass jede Nachricht wenigstens einmal aufgezeichnet wurde, hinreichend hoch. Selbst wenn so nicht jede Nachricht gespeichert wird, kann dennoch ein vollständiges Bild des verteilten Systems erzeugt werden.

Eine weitere Fehlerquelle könnte im VLAN-Mirroring liegen. Will Prozess / Rechner A mit B kommunizieren, so sendet er die Nachricht über die Netzwerkkarte (NIC, Network Interface Card). Die Nachricht wird über den Switch an B weitergeleitet, mit aktiviertem Mirroring auch an den Capture PC. Es ist nicht auszuschließen, dass der Switch die Weiterleitung an den Capture PC zeitversetzt vornimmt, d.h. erst die Nachricht an B ausliefert, dann (eventuell sogar nachdem etwas Zeit vergangen ist), die Nachricht an den Capture PC weiterleitet. Eine Rekonstruktion des eigentlichen Zeitpunktes, wann die Nachricht versendet wurde bzw. wann sie ankommt, ist nicht möglich. So können sich Probleme bei den Algorithmen der Prozesssicht und der Abhängigkeitsgraphen ergeben, da diese Algorithmen auf Zeit basieren um einen Zusammenhang von einem Sendeereignis zu einem Empfangsereignis zu erkennen.

Somit können weder Empfangs- noch Sendeereignisse genau terminiert werden, beide erhalten für eine Nachricht den gleichen Zeitstempel, wenn sie den Capture PC erreicht. Eine Trennung der beiden Ereignisse ist nicht mehr möglich. Würden auf allen Computern die Nachrichten aufgezeichnet, müsste eine sehr genaue Uhrensynchronisation durchgeführt werden. Eine perfekte Synchronisation ist nicht möglich, weshalb auch hier Fehler entstehen würden.

8 Ausblick

In zukünftiger Forschung könnten weitere Modelle oder Ansätze ausgewertet werden, bspw. könnte die Verfügbarkeit von Kanten unter

der Zuhilfenahme von historischen Daten untersucht werden. Dabei könnte es genügen zu zählen wie viele Pakete gesendet worden sind, und wie viele beim Empfänger tatsächlich angekommen sind.

Mit Multigraphen, bei denen jeder Knoten einen Rechner (oder Prozess) darstellt, und jede Kante eine Kommunikation zwischen zwei Knoten über einen bestimmten Port darstellt, könnte es möglich sein abzuschätzen, wie viele Ausgangskanten ein Rechner hat, d.h. von wie vielen anderen Rechnern dieser benötigt wird. Rechner mit vielen Kanten sind offensichtlich im System sehr wichtig, und sollten durch StandBy-Rechner oder andere Fehlertoleranzmaßnahmen gesichert sein.

Anhand der Länge von Paketen kann untersucht werden, auf welchen Wegen durch das System wie viele Daten versendet werden. Die Wege mit besonders hoher Last sind für das System offenbar sehr wichtig, und sollten daher besonders verstärkt werden.

Anhand der Informationen, wann eine Anfrage eingetroffen ist, und wann eine Antwort gesendet wurde, d.h. an den gleichen Port des Anfragenden, kann es möglich werden Bearbeitungszeiten im System abzuschätzen und die langsamsten Knoten zu identifizieren. Diese Knoten könnten dann durch stärkere Hardware ersetzt werden, oder die Software kann restrukturiert werden und ggf. auf mehrere Prozesse aufgeteilt werden.

Die Auswertung der in Abschnitt 7.3 aufgeführten Graphen mit dem Brückenerkennungsverfahren ist zwar formal korrekt[3], aber für die Einschätzung der Verlässlichkeit des Systems wenig hilfreich. Es müssten geeignetere Verfahren gefunden werden um diese Modelle auswerten zu können. Denkbar wäre die Auswertung der Abhängigkeiten zu Fehlerbäumen.

Ein weiteres Forschungsfeld könnte die Erkennung von Prozessen über IP und Port sein. Hier wurde versucht über einen zeitlichen Schwellwert zu erkennen, ob eine gesendete Nachricht mit einer gerade empfangenen Nachricht zusammenhängt. Es könnte untersucht werden, ob es genauere Methoden dafür gibt; eventuell muss dabei auch auf Betriebssystem-Datenstrukturen zurückgegriffen werden.

9 Fazit

In diesem Artikel wurde gezeigt, wie ein verteiltes System durch die Auswertung von aufgezeichneten Kommunikationsinformationen automatisiert modelliert und auf Single Points of Failure untersucht werden kann. Es wurden drei verschiedene Detailgrade beleuchtet: von einer recht ungenauen Sicht (Topologiegraph) über eine genauere (Prozessgraph), bis später Richtung an Abhängigkeiten zwischen Prozessen angetragen werden konnten (Abhängigkeitsgraphen).

Die Topologiegraphen können ebenfalls genutzt werden um zu erkennen, welche Rechner miteinander kommunizieren. Für die Analyse der Verlässlichkeit eines verteilten Systems genügt das aber noch nicht. Die Prozesssicht führt einige Fehler in das Modell ein, dafür kann damit eine genauere Sicht auf das verteilte System erlangt werden. Die Abhängigkeitsgraph-Methode führt, mit

[3] unter Lockerung des Artikulationspunktbegriffs

dem richtigen Schwellwert dazu, dass Wege betrachtet werden können, die eine Anfrage genommen hat, ohne dabei die Nachrichten verändern zu müssen.

Es wurde mit der Brücken- und Artikulationspunkterkennung ein Verfahren dargestellt, mit dem die Auswirkungen von Ausfällen analysiert werden können. Damit ist es möglich auch dynamische Systeme zur Laufzeit zu evaluieren und zu erkennen, welche Rechner oder Prozesse im verteilten System besonders kritisch sein könnten. Das Verfahren ist für die Abhängigkeitsgraphen nicht sinnvoll einsetzbar.

Literatur

1. Buzacott, J.A.: A recursive algorithm for directed-graph reliability. Networks 13(2), 241–246 (1983)
2. Fonseca, R., Porter, G., Katz, R.H., Shenker, S., Stoica, I.: X-trace: A pervasive network tracing framework. In: Proceedings of the 4th USENIX Conference on Networked Systems Design & Implementation. pp. 20–20. NSDI'07, USENIX Association, Berkeley, CA, USA (2007), http://dl.acm.org/citation.cfm?id=1973430.1973450
3. Schmidt, J.M.: A simple test on 2-vertex- and 2-edge-connectivity. Information Processing Letters 113(7), 241 – 244 (2013), http://www.sciencedirect.com/science/article/pii/S0020019013000288
4. Uhle, J.: On Dependability Modeling in a Deployed Microservice Architecture. Master's thesis, Universität Potsdam (June 2014)

Nicht-blockierende Implementierung von verketteten Listen mit Hilfe der Intel® Transactional Synchronization Extension

Thomas Jakobs

TU Chemnitz, Fakultät für Informatik, Professur Praktische Informatik, Chemnitz
thomas.jakobs@informatik.tu-chemnitz.de

Zusammenfassung. Synchronisation ist eine Methodik für den Zugriff auf kritische Bereiche, die die Effizienz von Datenstrukturen mit parallelen Zugriffen beeinträchtigen kann. Meist werden zur Synchronisation Mutex-Variablen oder atomare Befehle eingesetzt. Darüber hinaus gibt es die Technik des transaktionalen Speichers, der durch die Einführung der Haswell Prozessorarchitektur viele Anwender erreichte. Dabei werden kritische Bereiche als Transaktion ausgeführt, wodurch eine Synchronisation erreicht wird. In dieser Arbeit wird gezeigt, wie durch die Verwendung der Prozessorerweiterung TSX der parallel Zugriff auf eine doppelt verkettete Liste umgesetzt werden kann. Das durchgeführte Experiment zeigt darüber hinaus, dass die entstandene Datenstruktur Parallelität effizienter umsetzt, als herkömmliche Lock basierte Methoden.

Schlüsselwörter: Transaktionaler Speicher, Doppelt verkettete Liste, Nicht-blockierend

1 Einleitung

Viele Algorithmen und Programme verlassen sich auf die Nutzung von Datenstrukturen, wie z. B. Listen, die für die Datenspeicherung eingesetzt werden. Solche Datenstrukturen müssen bei der Parallelisierung ebenfalls für den parallelen Gebrauch angepasst werden. Meist erfolgt eine solche Anpassung durch die Verwendung von Locks bzw. Mutex-Variablen. Performantere Versionen der Datenstrukturen können jedoch entworfen werden, wenn nicht-blockierende Synchronisationsmethoden verwendet werden. Viele solcher Datenstrukturen werden bereits durch die Nutzung von atomaren Befehlen von Herlihy & Shavit [1] vorgestellt. Dabei ist es jedoch nicht immer möglich eine Datenstruktur in eine solche Form zu überführen oder dabei alle Eigenschaften der Datenstruktur beizubehalten.

Eine weitere Möglichkeit Datenstrukturen in eine nicht-blockierende Form zu überführen ist die Nutzung eines transaktionalen Speichermodells. Dieses Modell kann entweder durch Software oder durch Hardware realisiert werden. Bei diesem Modell werden Operationen auf gemeinsamen Speicher als Transaktion dargestellt. Aus dem Bereich Datenbanken bekannte Transaktionen erfüllen die folgenden Eigenschaften:

- **Atomicy:** Eine Transaktion wird im Gesamten ausgeführt oder gar nicht.
- **Consistency:** Ist die Transaktion abgeschlossen liegt ein konsistenter Zustand vor.
- **Isolation:** Eine Transaktion darf keine andere Transaktion beeinflussen oder durch eine andere Transaktion beeinflusst werden.
- **Durability:** Ist die Transaktion abgeschlossen sind die Änderungen permanent.

Bei transaktionalem Speicher kann durch diese Eigenschaften erreicht werden, dass sich die Veränderung von mehreren Daten als ein Schritt sichtbar wird. Dies entspricht dem gleichen Verhalten, dass bei der Anwendung auf ein Datum mit atomaren Operationen erreicht wird.

Mit der erstmaligen Einführung von Prozessoren der Haswell Generation, wurden Möglichkeiten bereit gestellt durch Transaktionen hardware-seitig auf Speicher zuzugreifen. Diese wurden unter dem Begriff *Transactional Synchronization Extension* (TSX) zusammengefasst. Die TSX Erweiterung der Prozessoren wurde mittlerweile weitgehend durch Firmware und BIOS Updates abgeschaltet, da ein Fehler gefunden wurde [4]. Dieser Fehler führt zu einem undefinierten Verhalten der CPU und kann nicht durch Software ausgeglichen werden. Intel® hat daraufhin die Abschaltung der Erweiterung auf allen verfügbaren CPUs bekannt gegeben. Diese Arbeit entstand vor bekannt werden des Fehlers, Intel® plant jedoch die Wiedereinführung von TSX mit der nächsten Prozessorgeneration. Aus diesem Grund können die hier gezeigten Beobachtungen weiter verwendet werden.

Ziel dieses Artikels ist es einen Einblick in die Technologie TSX zu bekommen und zu untersuchen, ob durch deren Nutzung parallele Programme effizienter abgearbeitet werden können. Dabei stehen nebenläufige Systeme mit geteiltem Speicher im Fokus, bei denen die Synchronisation mehrerer Threads durch Transaktionen ersetzt werden können.

Im folgenden Abschnitt wird TSX näher vorgestellt. Im Anschluss daran werden in Abschnitt 3 die Grundlagen für das durchgeführte Experiment erläutert. In Abschnitt 4 werden die Messergebnisse des Experiments vorgestellt und diskutiert. Der Artikel schließt mit einem Fazit und Ausblick auf zukünftige Herausforderungen.

2 Transactional Synchronization Extension (TSX)

Die in den Haswell Prozessoren implementierte TSX stellt zwei verschiedene Mechanismen für die Nutzung von Transaktionen zur Verfügung. Hardware Lock Elision (HLE) wird durch den Compiler automatisch auf übersetzten Code angewendet und Restricted Transactional Memory (RTM) kann durch den Programmierer explizit aufgerufen werden. Bei beiden angebotenen Mechanismen wird der kritische Bereich als Transaktion ausgeführt. Dazu werden Schreiboperationen durch den Cache-Buffer zwischengespeichert, und gleichzeitig durch das Cache-Coherence Protokoll Schreiboperationen auf die gleichen Cache-Zeilen

abgefangen. Treten Schreiboperationen auf die gleichen Cache-Zeilen durch andere Prozessorkerne auf, so wird die Transaktion abgebrochen [5]. Durch dieses Verfahren werden im Erfolgsfall keine Warte- oder Sperroperationen nötig. Im Folgenden sollen die beiden angebotenen Mechanismen kurz vorstellt werden.

2.1 Hardware Lock Elision (HLE)

HLE kann implizit genutzt werden. Dabei werden die Aufrufe an Mutex-Variablen vom Compiler in entsprechende HLE-Befehle umgewandelt.

Versucht während der Ausführung ein Programm diesen Mutex zu akquirieren, wird stattdessen eine Transaktion gestartet. Wird der kritische Bereich nur von einem Thread zur gleichen Zeit bearbeitet, so kann die Transaktion erfolgreich abgeschlossen werden und das Akquirieren des Mutex ist nicht notwendig. Tritt ein zweiter Thread zusätzlich in diesen kritischen Bereich ein, so wird die Transaktion des ersten Threads abgebrochen und beide Threads nutzen die Mutex-Variable statt einer Transaktion.

Diese Befehle arbeiten mit einem Präfix für Maschinenbefehle, der auf Prozessoren mit fehlender HLE Kompatibilität ignoriert wird. Ein Prozessor mit fehlender HLE Kompatibilität wird dementsprechend keine Transaktion starten, sondern stattdessen den Mutex direkt sperren.

2.2 Restricted Transactional Memory (RTM)

Bei der Nutzung von RTM gibt der Programmierer explizit an, an welcher Stelle Transaktionen in welcher Form ausgeführt werden sollen. Dazu stehen die vier Funktionen `_xbegin`, `_xtest`, `_xabort` und `_xend` zur Verfügung.

Mit der Funktion `_xtest` kann getestet werden, ob der Prozessor gerade eine Transaktion durchführt. Mit `_xbegin` wird eine Transaktion gestartet, wobei der Status als `unsigned int` zurück gegeben wird. Ist der Status 0xFFFFFFFF (bzw. -1) so wurde die Transaktion erfolgreich gestartet.

Nach dem erfolgreichen Starten einer Transaktion wird der nachfolgende Code als Transaktion ausgeführt, bis die Funktionen `_xabort` oder `_xend` aufgerufen werden oder einer der im Folgenden beschriebenen Gründe zum Abbruch führen.

Die Funktion `_xend` beendet eine Transaktion als erfolgreich, wodurch die Änderungen für andere Prozessorkerne sichtbar werden.

Wird `_xabort` aufgerufen oder ein anderer Prozessorkern greift beispielsweise auf die genutzten Daten zu, wird die Transaktion abgebrochen. Dies hat zur Folge, dass sämtliche Änderungen der Transaktion verworfen werden, was sich auch auf den Programmfortschritt bezieht. Daraus folgt, dass der erste Befehl nach Abbruch einer Transaktion die Rückgabe des Status-Wertes aus `_xbegin` ist. Dieser Status ist nun jedoch verschieden von 0xFFFFFFFF. Somit lässt sich nur an diesem Status-Wert erkennen, ob eine Transaktion erfolgreich gestartet wurde oder während der Ausführung abgebrochen wurde.

Der Funktion `_xabort` muss ein Wert übergeben werden, der mit in den `status`-Wert codiert wird. Diese Codierung kann, wie in Tabelle 1 dargestellt,

in mehrere Informationsbits aufgeschlüsselt werden, um daraus Hinweise zum weiteren Vorgehen zu erhalten. Wird eine verschachtelte Transaktion abgebrochen beginnt die Neuausführung immer am äußersten aufgerufenen `_xbegin`.

EAX register bit	**Meaning**
0	Set if abort caused by XABORT instruction.
1	If set, the transaction may succeed on a retry. This bit is always clear if bit 0 is set.
2	Set if another logical processor conflicted with a memory address that was part of the transaction that aborted.
3	Set if an internal buffer overflowed.
4	Set if a debug breakpoint was hit.
5	Set if an abort occured during execution of a nested transaction.
23:6	Reserved.
31:24	XABORT argument (only valid if bit "0" set, otherwise reserved).

Tabelle 1. Aufschlüsselung des Rückgabewertes der Funktion `_xbegin` bei Abbruch einer Transaktion. Quelle: [2]

Listing 1.1 zeigt eine minimalistische Möglichkeit wie eine Transaktion in RTM programmiert werden kann. Dabei wird nach dem Aufruf von `_xbegin` die Variable `status` dahingehend überprüft, ob es sich um eine gestartete Transaktion handelt. Ist dies der Fall, wird in Zeile **??** eine Abbruchbedingung überprüft, die wenn zutreffend, die Transaktion mit `_xabort` abbricht. In Zeile **??** wird die Transaktion erfolgreich beendet. Schlägt die Transaktion aus einem der oben genannten Gründe fehl, so beginnt die Ausführung erneut in Zeile **??** durch die Zuweisung an `status`. Hierbei wird die Funktion `_xbegin` nicht erneut aufgerufen, sondern lediglich der Rückgabewert verändert. Daraufhin schlägt der Vergleich in Zeile **??** fehl und der `else`-Zweig wird betreten. Dieser sollte einen Fallback-Code enthalten der ohne Transaktionen ausgeführt werden kann. Ein genaues Beispiel mit Code und Erklärungen findet sich unter [6].

Einige Befehle können innerhalb einer Transaktion nicht verwendet werden, da sie zu einem Abbruch der Transaktion führen. Die entsprechenden Prozessoroperationen, die einen solchen Abbruch hervorrufen, werden in [3] beschrieben. Insbesondere ist dies die Funktion `_xabort`. Weiterhin führen Heap-Managementoperationen oder Ein- und Ausgabeoperationen, sowie deren versteckte Aufrufe zu einem Abbruch. Dies kann der Fall sein, wenn eine Bibliotheksfunktion verwendet wird oder ein Smartpointer verändert wird.

3 Experiment

Als Vergleich zur transaktionalen Datenstruktur wird eine naive Implementierungen der doppelt verketteten Liste genutzt. Diese Implementierung wird da-

```
unsigned int status = _xbegin();
if(status == 0xFFFFFFFF) {
  if(stopping condition)
    _xabort(1);
  // Code
  _xend();
} else{
  // Fallback-Code that is not transactional
}
```

Listing 1. Minimales Makro zur Ausführung einer Transaktion mit RTM

bei selbst erstellt, um beide Datenstrukturen ausschließlich in Bezug auf die angewendeten Veränderungen durch TSX zu untersuchen.

Als fiktives Problem zur Nutzung der doppelt verketteten Liste wurde ein paralleler Einfüge-Such-Lösch-Bot (ESL-Bot) gewählt. Für die durchgeführten Tests wird die Liste mit zufälligen Elementen vorab befüllt. Bei der Durchführung der Tests wird zufällig eine der drei Operationen Einfügen, Suchen oder Löschen ausgewählt und diese dann mit einer Zufallszahl aufgerufen.

3.1 Vergleichsstruktur

Hierbei wird beim Eintritt in die entsprechende Methode ein Mutex gesperrt, der für die gesamte Datenstruktur gilt. Dadurch werden die Zugriffe auf die gesamte Datenstruktur sequentialisiert.

Um für eine Vergleichbarkeit zu sorgen, gibt es zwei Implementierungen. Die erste Implementierung basiert auf Raw-Pointern, die durch die Programmiersprache zur Verfügung gestellt werden. Die entwickelten transaktionalen Versionen benötigen jedoch den Einsatz von C++11 Smartpointern. Diese werden in einem zweiten Basisalgorithmus verwendet.

Bei der doppelt verketteten Liste werden Elemente sortiert in die Liste eingefügt, falls noch kein Element mit diesem Inhalt enthalten ist. Beim Suchen wird auf den Inhalt des Listenelements abgeprüft und eine Kopie dessen zurückgegeben. Das Löschen entfernt ein vorhandenes Listenelement aus der Liste und gibt dessen Speicher frei. Bei der Variante mit Smartpointern wird dabei das Element nur ausgehängt, da eine Löschung automatisch erfolgt, sobald kein weiterer Zeiger mehr auf das Element verweist.

3.2 Transaktionale Datenstruktur

In diesem Teil wird zur Beschreibung der Datenstruktur ein transaktionaler Bereich genutzt, bei dem vorausgesetzt wird, dass er auf jeden Fall erfolgreich abschließt.

Bei der Verwendung von Transaktionen ist darauf zu achten, dass die transaktionalen Bereiche möglichst kurz gehalten werden, um die Wahrscheinlichkeit einer Fertigstellung zu erhöhen [3]. Aus diesem Grund ist es nicht zielführend die gesamte Funktion in einen transaktionalen Bereich umzuwandeln. Durch die

Zerstückelung der Funktion entstehen Bereiche, in denen von anderen Threads Daten manipuliert werden können, die für den weiteren Verlauf dieser Funktion notwendig sind. Somit ist es nicht möglich eine transaktionale Implementierung der Funktionen auf Basis von Raw-Pointern und ohne weitere Synchronisationsmechanismen zu entwerfen.

In [1] wird in Kapitel 9 an eine nicht-blockierende Implementierung einer Liste herangeführt. Da die dort vorgestellten Prinzipien für die Sprache Java, jedoch nicht für C/C++ funktionieren, werden im Folgenden die Prinzipien übertragen angewendet.

Zu Beginn ist darauf zu achten, dass Elemente, die von einem Thread aus der Liste entfernt werden sollen zu diesem Zeitpunkt noch von anderen Threads referenziert sein können. Dazu wird das Element aus der Liste ausgehängt und später durch den Smartpointer (Original Garbage-Collector) entfernt.

Die verwendeten Smartpointer sind im Header `memory` der Standardbibliothek definiert. Es werden `std::shared_ptr` und `std::weak_ptr` verwendet. Die Vorwärtszeiger der Liste und alle lokal benötigten Zeiger sind `std::shared_ptr`, die das oben beschriebene Verhalten bieten. Für die Rückwärtsverzeigerung wird durch `std::weak_ptr` umgesetzt, deren Referenz bei anderen Smartpointern nicht mitgezählt wird. Um diese `std::weak_ptr` zu dereferenzieren, müssen sie vorher in einen `std::shared_ptr` umgewandelt werden.

Da Elemente, die aus der Liste ausgehängt sind, weiterhin referenziert werden können, ist es darüber hinaus notwendig eine Möglichkeit anzubieten mit der darauf zugreifende Funktionen erkennen können, dass dieses Element eigentlich nicht länger in der Liste ist. Hierzu wurde ein Flag eingeführt, dass auf wahr gesetzt wird sobald das Element als gelöscht gilt [1]. Dadurch ist es nicht notwendig die Liste von vorne zu durchlaufen, wenn ein Suchlauf auf ein gelöschtes Element trifft. Ein Thread kommt beim iterieren in jedem Fall in die Originalliste zurück, ohne dass eine alternative Behandlung notwendig wäre.

Herlihy & Shavit [1] beschreiben weiterhin, dass es zu einem Problem kommen kann, wenn zwei Threads aufeinanderfolgende Elemente aus der Liste entfernen möchten. Dies ist dabei durch eine atomare Operation gelöst wird, die hier Ansatzpunkt für die Nutzung von transaktionalen Bereichen ist. Durch die Nutzung des transaktionalen Bereichs können diese Löschoperationen entweder nacheinander stattfinden, oder die Transaktionen werden abgebrochen und erneut ausgeführt.

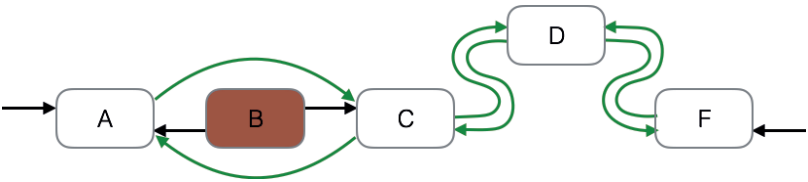


Abb. 1. Umbiegen der Pointer beim Einfügen und Löschen eines Elements in der Liste.

Für das Traversieren der Liste wird eine Schleife genutzt, die thematisch getrennt wird. Der Schleifenkopf enthält lediglich ein Flag, das mit wahr initialisiert wird und auf falsch gesetzt wird sobald das Ziel der Traversierung oder das Listenende erreicht wurde. Der Schleifenkörper wird als Transaktion ausgeführt und enthält die eigentliche Schleifenbedingung und das Weitersetzen des Pointers. Die Transaktion macht dabei die Prüfung der Bedingung und das Weitersetzen des Pointers in einem geschützten Bereich. Durch die Smartpointer, wird das referenzierte Element nicht außerhalb der Transaktion ungültig.

Beim Traversieren wird die Liste von vorne nach hinten durchlaufen, bis eine angegeben Position oder das Ende der Liste erreicht wird. Das Suchen in der Liste kann direkt über die Traversierung abgebildet werden und benötigt keine weitere Programmlogik.

Beim Einfügen wird die richtige Position des Elements durch Traversieren gesucht, das Element außerhalb des transaktionalen Bereichs erstellt und dann mittels einer Transaktion eingefügt. Dabei werden wie in Abbildung 1 dargestellt die Pointer der benachbarten Elemente umgebogen. Dies findet dabei als Transaktion statt, wodurch alle oder kein Pointer umgebogen werden. Da sich verschiedene Threads beliebig verschränken können, ist es nicht zulässig ein Element einzufügen, wenn eines der Nachbarelemente zum Löschen markiert ist. In einem solchen Fall muss die Liste erneut durchlaufen werden, da sonst das Element unter Umständen außerhalb der Liste einfügt wird.

Beim Löschen eines Elements, wird es über die Traversierung gesucht, und wenn vorhanden aus der Liste ausgehängt. Ist das Element in der Liste enthalten, so wird es markiert und die eingehenden Pointer der Nachbarelemente in einer Transaktion um das Element herumgebogen. Die ausgehenden Pointer werden wie in Abbildung 1 gezeigt nicht verändert. Beim Löschen eines Elements wird zu Beginn ein Dummy-Pointer erstellt, der auf das zu löschende Element zeigt. Wird das Element nicht mehr referenziert, so wird es bei Verlassen des Gültigkeitsbereichs des Dummy-Pointers gelöscht. Auch das Löschen ist nicht zulässig, wenn das Element oder eines der Nachbarelemente markiert ist, wobei dann ebenfalls die Traversierung wiederholt wird.

Bei diesen Operationen werden stets Sonderfälle wie erstes, letztes oder einziges Element der Liste vorher abgehandelt.

3.3 Makro für transaktionalen Bereich

Da bei den vorliegenden Programmen TSX getestet werden sollte, wird nach dem ersten Fehlschlag einer Transaktion nicht auf einen Fallback-Code ausgewichen. In Listing 1.2 wird die Transaktion so lange wiederholt, bis sie erfolgreich abgeschlossen werden kann und das Flag `done` nur dann `true` enthält.

Ein auftretendes Problem besteht jedoch darin, dass beim Abbruch der Transaktion durch einen Konflikt, der durch einen anderen Prozessorkern verursacht wird, beide Prozessorkerne die Transaktion abbrechen. Direkt im Anschluss daran versuchen beide erneut die Transaktion auszuführen, woraufhin sie sich mit hoher Wahrscheinlichkeit erneut stören. Obwohl die Spezifikation [3] an dieser

```
1  bool done = false;
2  unsigned int status;
3  while(!done) {
4    status = _xbegin();
5    if(status == 0xFFFFFFFF) {
6      // Code
7      done = true;
8      _xend();
9    } else if(status & 4) {
10     std::this_thread::yield();
11   }
12 }
```

Listing 2. Erweitertes transaktionales Makro ohne Nutzung eines Fallback-Codes mit Konfliktlösungsstrategie

Stelle angibt, dass es bei solchen Konflikten einen Gewinner gibt, der die Transaktion erfolgreich abschließt, konnte dieses Verhalten mehrfach beobachtet werden.

Eine faire und effiziente Möglichkeit diese zufällige Unterscheidung einzubringen ist die Nutzung des Betriebssystems. Hierzu kann die verbleibende Zeitscheibe der beteiligten Threads an die Betriebsmittelverwaltung zurück gegeben werden. Der Vorteil gegenüber einer Methode mit Zufallszahlen ist hierbei, dass keine weiteren Berechnungen durchgeführt werden, sondern die zufällige Größe der verbleibenden Zeitscheibe genutzt wird.

In Listing 1.2 wird überprüft, ob das Bit 2 des EAX Registers (vgl. Tabelle 1) gesetzt ist, welches angibt, ob der Abbruch durch einen Konflikt hervorgerufen wurde. Die Funktion `yield` übernimmt die oben beschriebene Funktionalität, und gibt die verbleibende Zeitscheibe dieses Threads an die Betriebsmittelverwaltung zurück. Durch dieses Vorgehen wird der auftretende Konflikt, der zu einem Life-Lock führen kann, über den Scheduler aufgelöst.

4 Diskussion

In diesem Abschnitt werden die drei implementierten Versionen miteinander verglichen. Dazu werden die Versionen mit unterschiedlichen Eingabeparametern ausgeführt. Die Ausführung fand auf einem Haswell i7-4770 Prozessor statt. Es wurde der Intel® Compiler auf einer Arch Linux Installation verwendet.

Im Folgenden werden die Performance-Eigenschaften der geschaffenen Datenstruktur diskutiert. Hierzu werden die Eigenschaften des Testalgorithmus anhand der folgenden Begriffe unterschieden:

- **Anzahl Threads:** Gibt die Zahl der Threads an, mit der das Testprogramm ausgeführt wird.
- **Wertebereich:** Bezeichnet die Menge an natürlichen Zahlen, die in die Liste eingefügt werden können. Ein höherer Wertebereich führt zu einer größeren Maximalanzahl von Elementen der Liste. In der Regel gilt dabei, dass die Liste in etwa halb so viele Elemente enthält wie der Maximalwert des Wertebereichs angibt.

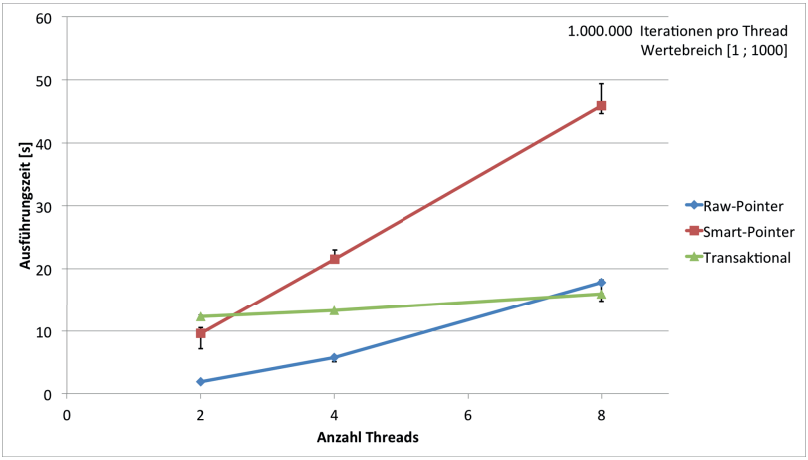


Abb. 2. Ausführungszeit des ESL-Bot bei gleicher Arbeit pro Thread mit unterschiedlicher Anzahl Threads.

– **Iterationen:** Die Anzahl der Iterationen gibt an, wie oft die Funktionen des ESL-Bot aufgerufen werden. Sie können sich dabei auf eine Anzahl pro teilnehmenden Thread oder eine Gesamtanzahl, die sich alle Threads teilen, beziehen.

Die gezeigten Graphen bestehen jeweils aus 20 Messwerten pro Datenpunkt, bei denen in jeder Gruppe das arithmetische Mittel angezeigt wird. Darüber hinaus wird die jeweilige Abweichung zum größten und kleinsten Wert der Messreihe durch einen Fehlerbalken dargestellt.

In Abbildung 2 wird der ESL-Bot mit der gleichen Anzahl an Iterationen pro Thread gestartet. Dies bedeutet, dass bei zwei Threads insgesamt zwei Millionen mal iteriert, und bei acht Threads entsprechend acht Millionen mal iteriert wird. Dabei ist zu erkennen, dass die beiden naiven Implementierungen wie erwartet einen linearen Anstieg mit der Gesamtanzahl der Elemente aufweisen, da hierbei sämtliche Operationen sequentialisiert werden müssen. Die transaktionale Implementierung hat hier nur einen leichten Anstieg der Ausführungszeit, da sich durch die Feinkörnigkeit der Transaktionen die Threads besser verzahnen können. Der leichte Anstieg wird dabei durch die höhere Wahrscheinlichkeit eines Konflikts und der daraus resultierenden Neuausführung hervorgerufen.

In Abbildung 3 wird der ESL-Bot mit der gleichen Anzahl an Gesamtiterationen gestartet. Dies bedeutet, dass bei zwei Threads, jeder Thread vier Millionen Iterationen ausführt, und bei acht Threads jeder Thread nur eine Million Iterationen. Zu beobachten ist, dass die Vergleichsversionen nicht skalieren und sogar mehr Zeit zur Abarbeitung benötigen, da dabei die Iterationen sequentialisiert werden und der Sequentialisierungsoverhead mit der Anzahl der Threads zu-

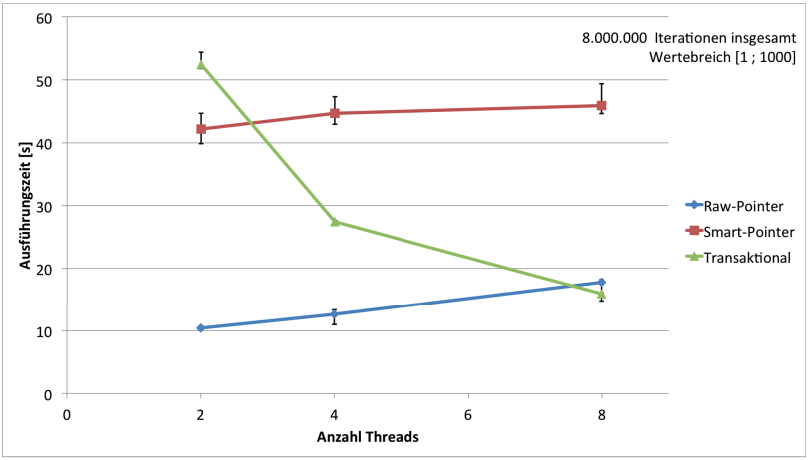


Abb. 3. Ausführungszeit des ESL-Bot bei gleicher Gesamtarbeit mit unterschiedlicher Anzahl Threads.

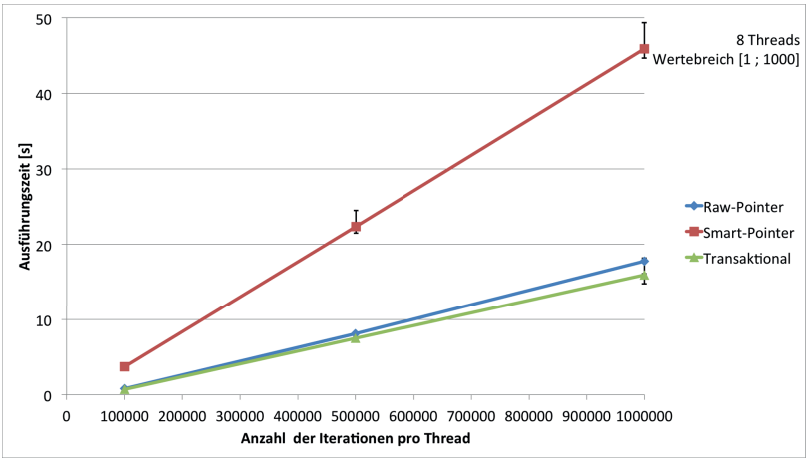


Abb. 4. Ausführungszeit des ESL-Bot bei unterschiedlicher Anzahl von Wiederholungen mit 8 Threads.

nimmt. Die transaktionale Variante skaliert dagegen gut. Jedoch tritt hier eine gewisse Sättigung des Speedup ein, wie anhand des Knicks im Graphen zu erkennen ist. Diese Sättigung ist ebenfalls auf die höhere Wahrscheinlichkeit eines Konflikts zwischen Prozessorkernen zurückzuführen.

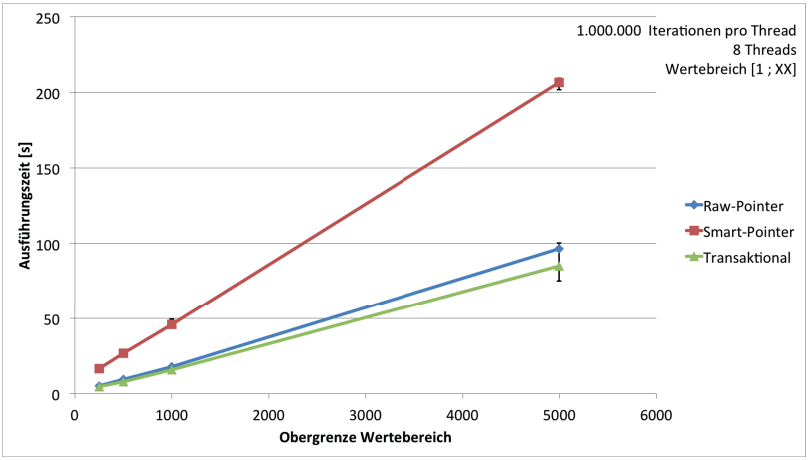


Abb. 5. Ausführungszeit des ESL-Bot bei unterschiedlich großen Wertebereichen mit 1 Million Iterationen und 8 Threads.

In Abbildung 4 sind die Laufzeiten in Abhängigkeit zu den durchgeführten Iterationen pro Thread dargestellt. Dabei ist klar zu erkennen, dass die Iterationsanzahl einen linearen Einfluss auf die Laufzeit bei allen drei Datenstrukturen ausübt und somit andere Messungen nicht beeinflusst. Die Steigung der Laufzeit ist dabei auf das mehr an Arbeit, dass jeder Thread an dieser Stelle auszuführen hat, zurückzuführen. Aus der größeren Steigung der Vergleichsstruktur lässt sich erkennen, dass der Overhead pro Iteration bei den Vergleichsstrukturen größer ist als bei der transaktionalen Implementierung.

In Abbildung 5 werden die Messergebnisse in Bezug auf den Wertebereich der Liste angezeigt. Der Wertebereich beginnt stets bei Eins und endet bei dem angegebenen Wert. Eine Vergrößerung des Wertebereichs wirkt sich somit, über die Länge der Liste, direkt auf die Laufzeit des Traversierens aus, was sich wiederum direkt auf die Gesamtlaufzeit auswirkt. Klar zu erkennen ist hier, dass die Länge des Traversierens einen stärkeren Einfluss auf die naiven Implementierungen hat, was sich wiederum aus der Sequentialisierung ergibt.

5 Fazit und Ausblick

Durch die Nutzung von TSX kann eine performante und skalierbare nicht-blockierende Datenstruktur erstellt werden. Der Geschwindigkeitsgewinn durch den Einsatz von TSX ist gegenüber der Vergleichsstruktur unter Nutzung der Smartpointer zu sehen. Berechnungen des Speedups verwenden jedoch die beste sequentielle Implementierung, weshalb der Gesamtgewinn nur gegenüber der Version mit Raw-Pointern betrachtet werden kann. In Zukunft kann die Anwendung weiterer Prinzipien aus zur Implementierung nicht-blockierender Datenstrukturen

die Nutzung von Smartpointern überflüssig machen. Durch diese Erweiterung könnte die Performance der transaktionalen Datenstruktur weiter gesteigert werden.

Die Auswirkung des transaktionalen Speicherzugriffs müssen für andere Datenstrukturen und Algorithmen ebenfalls evaluiert werden, um den praktischen Nutzen der Erweiterung einschätzen zu können. Generell eignet sich die Nutzung von Transaktionen zur Synchronisierung vor allem bei Herausforderungen, die nicht durch die Anwendung atomarer Befehle gelöst werden können. Sollen solche Herausforderungen dennoch eine höhere Performanz erreichen als eine Mutex-Variablen basierte Lösung bietet TSX hierbei eine einfache Möglichkeit der Programmierung.

Dies führt zu dem Schluss, dass TSX die Abarbeitung paralleler Programme beschleunigen kann. Jedoch müssen verschiedene Rahmenbedingungen beachtet werden, die den möglichen Vorteil wiederum abschwächen können. Die Nutzung einer Datenstruktur ist dabei nicht die günstigste Variante, da hierbei hauptsächlich Heapelemente verwendet werden, die in Erstellung, Nutzung und Löschung eine gesonderte Behandlung benötigen. Ein größerer Effizienzgewinn könnte erzielt werden, wenn die Wahrscheinlichkeit eines gleichzeitigen Zugriffs auf Daten sehr gering ist.

Literatur

1. Herlihy, M., Shavit, N.: The Art of Multiprocessor Programming. Elsevier, Amsterdam (2012)
2. Intel® Corporation: Restricted Transactional Memory Overview (12 2013), `https://software.intel.com/en-us/node/582937`
3. Intel® Corporation: Transactional Synchronization Extensions (Intel® TSX) Programming Considerations (12 2013), `https://software.intel.com/en-us/node/582935`
4. Intel® Corporation: Intel® Xeon® Processor E3-1200 v3 Product Family (1 2016), `http://www.intel.com/content/dam/www/public/us/en/documents/specification-updates/xeon-e3-1200v3-spec-update.pdf`
5. Rajwar, R.: Going Under the Hood with Intel's Next Generation Microarchitecture Codename Haswell (11 2012), `https://qconsf.com/sf2012/dl/qcon-sanfran-2012/slides/RaviRajwar_GoingUnderTheHoodWithIntelsNextGenerationMicroarchitectureCodenameHaswell.pdf`
6. Wooyoung, Kim: Fun with Intel® Transactional Synchronization Extensions (12 2013), `http://software.intel.com/en-us/blogs/2013/07/25/fun-with-intel-transactional-synchronization-extensions`

Konzeption und Implementierung einer Sprachsteuerung für Roboter

Kevin Kendzia

Technische Universität Chemnitz
kevin.kendzia@gmail.com

Abstract

In diesem Paper werden zuerst verschiedene Grundlagen für eine Spracherkennung auf mobilen Betriebssystemen betrachtet und im Zweiten wird eine Versuchsreihe beschrieben, wodurch optimale Einstellungen der Pocketsphinx-Einstellungen erzielt werden sollen. Vor allem wird die Empfindlichkeit der Spracherkennung im Hinblick auf Umgebungsgeräusche betrachtet und inwieweit sich die verschiedenen Werte auf diese auswirkt.

Einleitung

Schon in den 1960er Jahren begann man an Spracherkennungssystemen zu Arbeiten. Damals war IBM ein Vorreiter der Technik, doch konnte das erste System "Shoebox" gerade einmal 16 Worte verstehen [1]. Die Anforderungen sind über die Jahrzehnte immer weiter gestiegen, da sich sowohl die Technik als auch die vorhandene Software weiterentwickelt hat. Vor allem im mobilen Bereich wird die Bedienung von alltäglichen Funktionen wie Termine eintragen, Nachrichten verschicken, Suchanfragen ausführen oder Musik abspielen durch eine Sprachsteuerung erheblich vereinfacht.

Auch die Forschung hat den mobilen Markt für sich entdeckt. Das Projekt Human-Robot Cooperation, kurz H-RoC, welches eine Zusammenarbeit der TU Chemnitz und der HTW Dresden ist, befasst sich mit der Steuerung eines mobilen Roboters in unstrukturierten Umgebungen, speziell Großbaustellen. Die Roboter agieren als Experten und sollen assistieren und beraten. Zur Interaktion mit den Roboter stehen den Menschen (in diesem Fall Arbeiter auf einer Baustelle) Mobilgeräte auf Android-Basis zur Verfügung.

Dieses Szenario stellt an die Implementierung einer Spracherkennungssoftware im Wesentlichen drei Bedingungen. Sie muss mobil, robust bei starken Hintergrundgeräuschen und für Befehlsfolgen (Command-and-Control) konfigurierbar sein. Für diese Anforderungen eignet sich das flexible Spracherkennungsframework CMU PocketSphinx ganz besonders. Voraussetzung dafür sind einerseits geeignete Grammatiken und nicht zuletzt auch eine passende Parametrisierung, die in dieser Arbeit vorgestellt werden.

Spracherkennung

Sprache ist eine sehr komplexe und natürlich vorkommende Erscheinung. Viele Menschen verstehen kaum wie sie entsteht und vor allem wie sie wahrgenommen wird. Dabei wird oft angenommen, dass Sprache nur aus Wörtern besteht, welche wiederum aus Tönen beziehungsweise Phonemen bestehen. In Wirklichkeit besteht sie aus einer Folge von akustischen Signalen und ist ein dynamischer Prozess ohne klar unterscheidbare Teile – dafür gibt es zu viele Faktoren. [2] Spracherzeugungsbeschränkungen existieren für viele verschiedene Sprachen mit unterschiedlichen Klängen. Trotzdem hat die Sprache ein generelles Muster und kann gut durch verschiedene Charakteristiken wie Lautstärke, Frequenzverteilung, Tonhöhe und Silbenrate beschrieben werden [3]. Für die richtige Erkennung der Sprache gibt es verschiedene Ansätze bei Pocketsphinx. So existiert neben einer phonetischen Suche auch eine N-Gramm-Suche, welche mit Wahrscheinlichkeiten arbeitet, die angeben, welches Wort als nächstes gesagt wird. Zusätzlich existiert eine grammatikalische Suche. Bei dieser Art der Erkennung ist die Reihenfolge der zu erkennenden Wörter durch eine vordefinierte Grammatik vorherbestimmt. Für den Anwendungsfall einer Befehlssteuerung scheint dieser Ansatz besonders geeignet. Der hohe Grad der Flexibilisierung von PocketSphinx erlaubt auch die Verwendung von eigens entwickelten und trainierten Akustikmodellen und somit eine individuelle Anpassung an unterschiedliche Einsatzzwecke.

Der PocketSphinx Spracherkennungsprozess

Das Erkennen eines Sprachmusters in einer Datenbank von Wörtern, Phrasen und Phonemen ist ein hochkomplexer Vorgang. Hierbei wird die Wellenform des Signals in Äußerungen („utterances“) geteilt, indem man bei jedem stillen Part das Signal abtrennt. Anschließend wird versucht das Gesprochene zu erkennen, indem jede mögliche Wortkombination mit der vorhandenen Audioaufnahme verglichen wird. Die am besten passende Kombination wird daraufhin ausgewählt. [2] Es gibt 4 Stufen um diesen Erkennungsprozess zu verwirklichen.

Die erste Stufe ist die Vorverarbeitung. Diese befasst sich mit der Abtastung, Filterung und Transformation des Signals und sorgt dafür, dass das kontinuierliche Audio-Signal in ein diskretes umgewandelt wird. In der Filterung werden unwichtige Frequenzen, sowie Störgeräusche herausgefiltert. Da neben dem zeitlichen Signal für die Spracherkennung auch die Frequenzanteile wichtig sind, werden diese mit Hilfe der Schnellen Fourier-Transformation (FFT) aus dem Signal erstellt. Abschließend werden diese in circa 10 Millisekunden große Frames geteilt, welchen 39 Nummern zugewiesen werden. Diese 39 Zahlen werden als Merkmalsvektor bezeichnet und repräsentieren das Gesprochene. Der Vorgang, um diesen Vektor zu generieren, ist unter stätiger Entwicklung. [2]
Das Konzept des Modells ist die zweite Stufe. Im Fall der Spracherkennung sind es sogar 3 Sprachmodelle die zum Einsatz kommen. Das Akustikmodell (acoustic model) beinhaltet Eigenschaften und Einstellungen für jedes der circa 4000 unabhängigen Senone. Ein Senon ist ein Detektor für Teile von Triphonen. Da der Aufwand zu groß wäre jedes Triphon mit dem Signal zu vergleichen, wird nur ein Teil jedes Triphons, das Senon, verglichen. Dabei ist die Abhängigkeit des Senons zum Kontext komplexer als den linken und rechten Nachbarn des Phonems zu betrachten. Es ist mehr eine komplexe Funktion, welche durch einen Entscheidungsbaum definiert wird. [2] Außerdem sind im Akustikmodell kontextunabhängige Eigenschaften für Phoneme enthalten.
Ein weiteres Modell ist das phonetische Wörterbuch (phonetic dictionary). In diesem werden die Phoneme auf Wörter abgebildet. Diese Abbildungen (Mappings) sind jedoch nicht unbedingt leistungsfähig, da oft nur eine geringe Anzahl an Aussprachen notiert werden kann. Gewöhnlich reicht es zwar aus, jedoch könnte diese Abbildung auch durch eine komplexe Funktion mit maschinellem Lernen realisiert werden.
Das letzte Modell, das Sprachmodell (language model), ist für die Beschränkung der Wortsuche zuständig. Es bestimmt, welches Wort auf die vorher erkannten Wörter folgen kann. Außerdem hilft es dem Matching unwahrscheinliche Wörter zu sondieren. Am meisten werden n-gram Sprachmodelle verwendet, in welchen statistisch hinterlegt ist, welches Wort auf welches und mit welcher Wahrscheinlichkeit folgt. Jedoch können auch Grammatiken, wie sie in dieser Arbeit verwendet werden, genutzt werden. Ziel dieses Modells, egal welche Technik verwendet wird, ist die Vorhersage von nachfolgenden Wörtern. Außerdem wird der Wortschatz auf die enthaltenen Wörter beschränkt. Dies kann vor allem ein Problem bei der Namenserkennung darstellen.
In Kombination ergeben diese drei Modelle die Spracherkennungsengine und die Basis des hier verwendeten Frameworks.

Formalen Sprachen können durch Grammatiken beschrieben werden. Diese beziehen sich nicht auf sprachliche Formen oder Funktionen im Satz, sondern nur auf den Bau der Sprache und deren Gesetzmäßigkeiten. Eine Grammatik G = (T, N, P, S) besteht aus einem 4-Tupel mit folgender Bedeutung:

T ist ein endliches Terminalalphabet, N ist eine endliche Menge von Nichtterminalsymbolen, P ist eine endliche Menge von Regeln (Produktionen) und S ∈ N ist das Startsymbol. Die von G erzeugte Sprache wird durch

$$L(G) = \left\{ w \in T^* \mid S \overset{P}{\Longrightarrow} w \right\}$$

beschrieben, das leere Wort mit dem griechischen Buchstaben Epsilon ε. [4]

Solche Grammatiken können auch in Spracherkennungssystemen eingesetzt werden. Dort wird oft das *Java Speech Grammar Format* JSGF verwendet, da es sehr schnell zu erlernen ist und trotzdem die Komplexität einer formalen Grammatik besitzt. Das Startsymbol, welches die Grammatik nach außen für das zu verwendende Programm bereitstellt, wird mit *„public“* gekennzeichnet. Nichtterminalsymbole sind in spitzen Klammern gestellt. Bei den Projektionsregeln steht immer genau ein Nichtterminal auf der linken Seite, welches zu einer Option auf der rechten Seite abgeleitet wird. Diese Optionen können eine Kombination aus Terminalen und Nichtterminalen sein und werden durch das Pipe-Symbol „|“ getrennt. Gruppierungen werden mit 2 runden Klammern „()“ gelöst und sind sowohl für bessere Übersicht einsetzbar, als auch um Auswahlmöglichkeiten zu markieren. Denn gerade bei den Optionen kann es sonst zu Missverständnissen kommen.

```
01 #JSGF V1.0;
02
03 grammar musikbeispiel;
04
05 public <startbefehl> = starte itunes | starte spotify |
06                          (musik <musikbefehl>)+;
07 <musikbefehl> = stop | start | bitte* vor+ | zurueck+;
```

Abbildung 1: Beispiel eines JSGF durch welches Sätze wie „starte itunes“ oder auch „musik start musik vor vor vor“ möglich sind.

So sind „Befehl1 | Befehl2 Befehl3 | Befehlt4“ und „(Befehlt1 | Befehlt2) (Befehl3 | Befehl4)“ beides valide Projektionen, bei denen allerdings unterschiedliche Sätze entstehen. Es ist aber auch möglich, dass ein normales Textsegment innerhalb der runden Klammern enthalten ist, welches durch andere Operatoren modifiziert wird. Eckige Klammern „[]“ werden verwendet um Optionen zu realisieren. Darin enthaltene Textsegmente können gesprochen werden, müssen aber nicht. Diese beiden Klammern können sich sowohl über Terminale, als auch über Nichtterminale erstrecken. Außerdem gibt es unterschiedliche unäre Operatoren. Die wichtigsten sind hierbei die Kleenesche Hülle (Kleene star) „*“ und die positive Hülle (Kleene plus) „+“. Beide Operatoren werden hinter einem Wort oder einer Gruppierung platziert. Der Stern bewirkt, dass das Objekt keinmal oder öfter wiederholt werden kann. Die positive Hülle dagegen sagt aus, dass das Objekt mindestens einmal, aber beliebig oft, gesagt werden muss.

Die <sil>-Tags stehen hier für silence, was Stille bedeutet. Man kann eines der 3 Wörter sagen, bevor wieder Stille erwartet wird. Dies ist die Grammatik der Befehlsoberfläche, welche exakt in dieser Auflistung durch die App umgesetzt wurde. Die Rechtecke beschreiben gesprochene Wörter oder Platzhalter für bestimmte Textbausteinen. Diese Textbausteine sind mit spitzen Klammern „<“ „>“ markiert.

Platzhalter	Einsetzungen
<stuff>	package, tool[s], visitor
<person>	the manager, jim, chris, the nearest coworker
<head>	control center, manager
<action>	to come to me, about the incident
<info>	Stored material, arriving packages, present coworkers
<location>	Central, warehouse, stockyard <number>
<number>	One, two, three, four

Tabelle 2: Platzhalter und ihre Ersetzungen in der Befehlsgrammatik

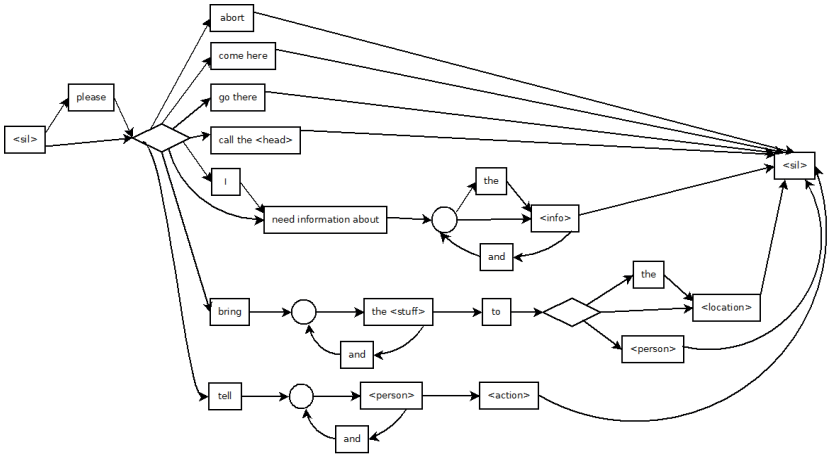


Abbildung 2: Grammatik der Befehlsoberfläche (Eigene Darstellung)

Die Pfeile zeigen die Auswahlmöglichkeit des Objektes an, welche teilweise von Rauten und Kreisen unterbrochen sind. Rauten sind Entscheidungsknoten, bei welchen man eines der nächsten Objekte auswählen kann. Dies gilt auch für 2 ausgehende Pfeile. Kreise sind Wiederholungsknoten, von denen eine Wiederholung der nächsten Objekte ausgehen kann. Auf diese Weise kann man über mehrere Sachen gleichzeitig Informationen erfragen.

Aus diesen Informationen können nun sinnvolle Sätze gebildet werden, welche die App versteht.

Beispielsatz 1: „I need information about stored material and present coworkers"

Folgende Route durch die Grammatik wird genommen, damit dieser Satz gebildet wird:

<sil> → i → need information about → <info> → and → <info> → <sil>

Beispielsatz 2: "please bring the tools and the package to jim"

Der Verlauf dieses Satzes ist etwas einfacher als der des Beispielsatzes 1:

<sil> → please → bring → the <stuff> → to → <person> → <sil>

Evaluation

Testbedingungen und Vorbereitungen

Der Versuchsaufbau fand in zwei unterschiedlichen Räumen statt. Ein hallarmer Raum und einer, in welchem viel Nachhall erzeugt wird. Da auf Baustellen auch in Gebäuden gearbeitet wird, in welchen es viel Hall gibt, sollte eine Lösung geschaffen werden, welche in beiden Umgebungen funktioniert. Bei schallreichen Räumen gibt es das Problem, dass das Echo das eigentlich gesprochene überlagern kann und somit zu falschen Ergebnissen führt.
Die zwei Räume sind beide ungefähr 20m² groß. Dabei wurde ein Glastisch an die Wand gestellt, auf welchem das Smartphone mit einem Stativ befestigt. Die Befestigung am Tisch wurde mit einem Saugnapf realisiert, welcher eine stabile und sichere Haltung bot. Es wurden zuerst Versuche ohne Lärm durchgeführt. Dabei gab es nur sehr
leise Nebengeräusche, welche zu vernachlässigen sind. Da jeder Mensch eine unterschiedlich kräftige Stimme hat, wurden die Befehle mit drei Lautstärkestufen eingesprochen: Normales Sprechen, lautes Sprechen und lautes Rufen. Nachdem die beiden Werte unter den Gesichtspunkten in der App angepasst wurden, wurde die Versuchsreihe erneut ausgeführt, diesmal mit Ton[1].
Ziel dieser Evaluation ist es, im technischen Sinn, die Werte des Recognizers an die Versuchsumgebung anzupassen[2]. Dabei wurden hauptsächlich die zwei Einstellungen *vad_threshold* und *kws_threshold* betrachtet. Die Ausgangswerte waren initial bei beiden Flags die Standardwerte.

Testergebnisse

Als erstes wurde der Einfluss des *kws_threshold* auf die Erkennung des Keywords getestet, indem 10 mal hintereinander „ok robot“ gesagt und die positive Reaktion durch Umschalten der Oberfläche gezählt wurde.
Dabei gab es kaum Unterschiede in der Erkennung zwischen den Werten „1.0f“, „0.5f“ und „0.1f“, welche auch nur durch unregelmäßige Aussprache von „ok robot“ zu erklären sind. Ab dem Wert „1e-5f“, welches in Dezimaldarstellung umgewandelt „0.00001“ ergibt, tritt eine Verbesserung der Werte ein. Hier ist ein eindeutiges Muster zu erkennen. Wenn man bei einem Abstand von 50 Zentimetern zu laut ist, verringert sich die Erkennungsrate, jedoch muss man ab einem Meter schon recht laut sprechen, damit die Spracherkennung funktioniert. Bei „1e-20f“ muss die Aussprache von „ok robot“ immer noch recht genau sein, damit die Erkennung funktioniert. Aus den Logs im Android Studio konnte aber geschlussfolgert werden, dass eine Sprachaktivierung stattfand. Außerdem wurde festgestellt, dass bei der Aussprache das O von „OK“ sehr wichtig ist, denn es darf nicht zu kurz sein, damit die Phrase erkannt wird. Grund

[1] Dabei kamen die Soundbeispiele von http://www.salamisound.de/baustelle-sounds und http://www.noisli.com/ zum Einsatz.

[2] Leider stand mir kein Schallpegelmessgerät zur Verfügung, womit die Auswertung objektiver gewesen wäre.

dafür ist der Eintrag „ok OW K EY“ im Wörterbuch. Hier sollte noch der Eintrag „ok(2) AA K EY“ eingefügt werden, da somit auch ein kurzes „o“ am Anfang akzeptiert wird. Ab 4 Metern Entfernung müssen auch die Befehle lauter ausgesprochen werden, damit die Erkennung ohne Probleme funktioniert.
Bei „1e-40f“ war die Erkennung des Keywords sehr gut. Jedoch konnte die Befehlsoberfläche mit undeutlicher Aussprache und sogar durch „ok gogog“ ausgelöst werden. Dies könnte zu einer ungewollten Aktivierung des Roboters führen. Schon das Abspielen eines englischsprachigen YouTube-Videos führte mit dieser Einstellung zur Aktivierung, was aufgrund der Sicherheit auf keinen Fall passieren darf. Somit muss ein Kompromiss zwischen guter Erkennung des Keywords und der Aktivierung durch falsch positive Ereignisse geben. Deshalb gab es bei den Versuchen mit dem *kws_threshold*-Wert „1e-40f“ keine verwertbaren Ergebnisse.

Abstand kws-Wert	0,5cm (normal / laut / rufen)	1m (normal / laut / rufen)	2m (normal / laut / rufen)	3m (normal / laut / rufen)	4m (normal / laut / rufen)
1.0f	0 / 0 / 1	0 / 0 / 0	-	-	-
0.5f	1 / 1 / 0	0 / 1 / 0	0 / 0 / 0	-	-
0.1f	1 / 2 / 0	1 / 0 / 0	0 / 0 / 0	-	-
1e-5f	3 / 4 / 1	1 / 2 / 0	0 / 0 / 1	0 / 0 / 0	-
1e-10f	3 / 4 / 2	7 / 3 / 1	2 / 1 / 0	1 / 0 / 0	0 / 0 / 0
1e-20f	7 / 6 / 3	4 / 8 / 4	3 / 7 / 4	4 / 3 / 5	0 / 1 / 3
1e-30f	9 / 7 / 8	8 / 10 / 6	6 / 8 / 8	7 / 8 / 8	2 / 6 / 8
1e-40f	10 / 9 / 9	-	-	-	-

Tabelle 2: Spracherkennung des Keyword Spottings

Mit der Einstellung *vad_threshold* = „2.0“ und *kws_threshold* = „1e-30f“ wurde der gleiche Test mit den gleichen Abständen durchgeführt, jedoch unter Einfluss von Lärm in 3 Lautstärkestufen: leise, mittel und laut. Dies soll die verschiedenen Lautstärkepegel auf der Baustelle simulieren. Hier wurden, neben der Keyphrase, fünf Befehle eingesprochen und ermittelt, ob der Lärm die Erkennung beeinflusst oder sogar die Deaktivierung des Aufnahmeprozesses unterdrückt.

Diese 5 Befehle sind:

- „come here“
- „call the control center“
- „i need information about the stored material and present coworkers”
- “bring the tools and the visitor to the nearest coworker”
- “tell the manager to come to me”

Abstand Lautstärke	0,5cm (normal / laut / rufen)	1m (normal / laut / rufen)	2m (normal / laut / rufen)	3m (normal / laut / rufen)	4m (normal / laut / rufen)
Leise	5 / 5 / 5	5 / 5 / 5	5 / 5 / 5	4 / 5 / 5	4 / 5 / 5
Mittel	3 / 5 / 5	3 / 5 / 5	2 / 5 / 5	2 / 3 / 4	0 / 0 / 0
Laut	5 / 5 / 5	0 / 2 / 4	0 / 0 / 0	-	-

Tabelle 3: Erkennung der Befehle unter Lärm ohne Hall im Raum

Bei leisen Umgebungsgeräuschen war die Erkennung noch fast ohne Folgen. Lediglich das „s“ bei „tools“ im 4. Befehl wurde nicht richtig erfasst. Bei der mittleren Lautstärke kam es schon im ersten Versuch zu Fehlern. Hier wurden ganze Teile von Befehlen nicht erfasst oder sogar umgeändert. Bei einem Meter Entfernung schlug schon jedes dritte Keyword Spotting bei normaler Sprechlautstärke fehl. Hier wurden 6 von 10 „ok robot“-Eingaben nicht erkannt. Ab 3 Metern wurde die Erkennung des Keywords auch beim lauten Rufen schlechter. 8 von 10 Versuche schlugen fehl und auch die Befehle wurden teilweise nicht mehr richtig erkannt. Der für das Keyword Spotting verantwortliche Wert *kws_threshold* konnte nicht noch weiter nach unten korrigiert werden, da es sonst zu viele falsch positive Ergebnisse entstehen (siehe Seite 36). Ab 4 Metern und mittlerer Lautstärke gab es keine Erkennung des Keywords mehr.

Trotz sehr lauter Umgebungsgeräusche, welche eine normale Gesprächslauterstärke übertönen, werden bei einem Abstand von 50 Zentimetern alle Befehle korrekt ausgeführt. Auch die Deaktivierung der Sprachaufnahme funktioniert, wenn auch mit leichten Verzögerungen. Deswegen wurde die Ruhezeit nach der Sprachaufnahme auf 30 Frames reduziert. Schon bei einem Abstand von einem Meter gab es nur noch vereinzelt positive Ergebnisse des Keyword Spottings, da
der Lärm die Sprache übertönt. Ab einer Distanz von zwei Metern reagierte die App nicht mehr auf Spracheingaben.
Der Test mit den Hintergrundgeräuschen wurde anschließend in einem nachhallenden Raum durchgeführt, jedoch konnten fast keine Veränderungen festgestellt werden. Dennoch waren diese Änderungen leicht negativ im Gegensatz zum Raum ohne Hall.

Abstand **Lautstärke**	0,5cm (normal / laut / rufen)	1m (normal / laut / rufen)	2m (normal / laut / rufen)	3m (normal / laut / rufen)	4m (normal / laut / rufen)
Leise	5 / 5 / 5	5 / 5 / 5	5 / 5 / 5	5 / 5 / 4	4 / 5 / 5
Mittel	3 / 5 / 5	3 / 4 / 5	2 / 4 / 4	2 / 3 / 4	0 / 0 / 0
Laut	5 / 5 / 5	0 / 2 / 3	0 / 0 / 0	-	-

Tabelle 4: Erkennung der Befehle und Lärm im nachhallenden Raum

Für weitere Tests wurde der *vad_threshold*-Wert von „2.0“ auf „1.0“ verringert, doch schon bei der leisesten Stufe von Lärm kam es zu Komplikationen. Zum einen wurde die Erfassung der Befehle beeinträchtigt, da Geräusche als Worte gedeutet wurden, und zweitens wurde die Spracherkennung nicht mehr deaktiviert, wodurch onResult() nicht mehr aufgerufen und keine Ergebnisse mehr angezeigt wurden. Dadurch kehrte die App auch nicht mehr auf den Startbildschirm zurück, da die Funktion onEndOfSpeech() nicht mehr aufgerufen wird.

Fazit

Die Ergebnisse der Evaluation zeigen gut, wie sich die eingestellten Werte verhalten. Der Defaultwert des *kws_threshold* ist für das Szenario des H-RoC zu hoch eingestellt. Selbst wenn man eine perfekte Aussprache hat, ist die Erkennung mehr zufällig als deterministisch. Der eingestellte Wert „1e-30f" bietet hingegen eine sehr gute Erkennung mit einem einzigen Fall von falscher Aktivierung. Jedoch gibt es noch einen kleinen Spielraum. Bei „1e-28f" wird die Erkennung etwas schlechter, dafür gibt es weniger falsch positive Ergebnisse. Um eine bessere Erkennung zu Gunsten von falsch positiven Ergebnissen zu haben, stellt man den Wert auf „1e-32f" oder höher. Der Standardwert des vad_threshold hingehen ist sehr gut gewählt. Die Erkennung funktionierte ohne eine Änderung sehr gut sowohl bei Stille als auch bei Lärm.
Die App bildet als Prototyp nun eine Grundlage für die weitere Forschung im Bereich der Spracherkennung in unstrukturierten Umgebungen.

Ausblick

Im Laufe der Appentwicklung habe ich viel getestet, vor allem mit dem deutschen Sprachmodell. Dabei habe ich versucht das von Voxforge bereitgestellte Modell in die Anwendung zu implementieren, damit die Befehle auch auf Deutsch ausgeführt werden können. Leider ist mir dies nur begrenzt gelungen, da das Akustikmodell nicht so gut an die genutzten Phoneme der eingepflegten Wörter meines Wörterbuchs angepasst war. Diese Anpassung kann im Laufe des Forschungsprojektes fortgeführt werden, indem das Akustikmodell trainiert wird. Weiterhin gibt es für PocketSphinx keine Out-of-Grammar Worterkennung. Das heißt, dass Wörter, welche nicht in der Grammatik vorkommen auch nicht erkannt werden. Dies ist von Nachteil, wenn mit Namen gearbeitet werden soll, da diese erst in die App implementiert werden müssen, bevor diese genutzt werden können. Eine Lösung dafür wäre eine grammatikfreie Erkennung, welche die Anwendungsfälle über die Semantik des gesprochenen Satzes erkennt.

Literaturverzeichnis

[1] IBM, „Pioneering Speech Recognition," [Online]. Available: http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/speechreco/transform/

[2] CMUSphinx, „Basic concepts of speech," 2015. [Online]. Available: http://cmusphinx.sourceforge.net/wiki/tutorialconcepts. [Zugriff am 14 11 2015]

[3] P. W. Lippte, Automatentheorie und formale Sprachen - Vorlesung, WWU Münster, 2009

[4] I. H. Witten, Principles of computer speech, 1984

Evaluation von Touch-Gesten zur Steuerung eines nicht-humanoiden Roboters

Anna Zywietz

TU Chemnitz, Medieninformatik
anna.zywietz@informatik.tu-chemnitz.de

Abstract. Die Interaktion zwischen Mensch und Roboter spielt nicht mehr nur in der Wissenschaft und Forschung eine große Rolle, auch im alltäglichen Kontext wird sie immer bedeutender. Dafür stellt der Einsatz direkter Interaktionsformen, wie die Eingabe durch Touch-Gesten, eine besondere Herausforderung dar. Um eine möglichst intuitive Eingabemöglichkeit zu gestalten, wurde eine Methodik zur Evaluation von Touch-Gesten zur Steuerung eines nicht-humanoiden Roboters konzipiert. So können ausgewählte Touch-Gesten auf verschiedene Parameter, die für deren spätere Implementierung bedeutend sind, untersucht werden.

Keywords: Human-Robot-Interaction, Touch Interfaces, Mobilgeräte

1 Einleitung

Touch-Gesten stellen eine weit verbreitete Eingabemethode dar, welche sich auch für die Mensch-Roboter-Interaktion eignet. Immer häufiger werden halb-autonome, nicht-humanoide Roboter im Arbeitsumfeld des Menschen benötigt und eingesetzt, sodass eine unkomplizierte und fehlerfreie Steuerungsmöglichkeit hilfreich ist. Hierfür gibt es bereits Ansätze, wie Steuerungen von Robotern auf mobilen Eingabegeräten konzipiert sein können. Bisher wurden jedoch in den meisten Fällen detaillierte Benutzungsoberflächen entwickelt, die dem Nutzer die Interaktionsmöglichkeiten mit dem Roboter durch den Einsatz von Knöpfen oder anderen Steuerungsmöglichkeiten vorgeben. Die dabei eingesetzten Gestaltungs- bzw. Steuerungselemente sind an das Konzept des Natural Mapping angelehnt: Darstellungen wie ein Lenkrad stellen natürlich angeordnete Elemente mit ihren entsprechenden Funktionen dar [No02]. Von einem auf Schaltflächen basierendem Interface zur Steuerung eines Roboters soll jedoch Abstand genommen werden, um stattdessen den Fokus des Nutzers auf die exklusive Eingabe von intuitiv angewendeten Touch-Gesten zu legen. Dafür benötigt der Nutzer als Eingabemöglichkeit lediglich das Display an sich. Zusätzliche Bedienelemente wirken in diesem Zusammenhang eher störend und würden dem Konzept der ausschließlichen Eingabe von Touch-Gesten widersprechen.

Obwohl die Anwendung von Touch-Gesten eine vielversprechende, ausdrucksstarke und natürlich erscheinende Interaktionsform darstellt, ist ein reines, durch Gesten gesteuertes System oft schwer zu durchschauen. Die Zahl der Eingabemöglichkeiten ist nicht ersichtlich und die präzise Ausführung solch dynamischer Bewegungen kann nur schwer richtig bewertet werden [Ol14]. Aus diesem Grund sollen einheitliche Parameter zur Eingabe von Touch-Gesten untersucht werden. Entscheidend für die Umsetzung einer Steuerung ist die Intuitivität von Touch-Gesten, die sich speziell in ihrer Merkbarkeit widerspiegelt. Durch diese kann der Nutzer die Eingaben leicht erlernen und wiederverwenden [Na13]. Das Ziel der Evaluation von Touch-Gesten ist es, geeignete Eingaben zu ermitteln, wodurch der Nutzer einen halb-autonomen, nicht-humanoiden Roboter ohne lange Anlernzeit problemlos steuern und anweisen kann.

2 Konzeptentwicklung

Zur Anwendung von Touch-Gesten als intuitive Steuerungsmöglichkeit bedarf es zuvor einiger Untersuchungen, um auszuwerten, welche Touch-Gesten sich im Kontext einer Robotersteuerung eignen. Aufgrund der Vielzahl an Anwendungsfällen und den daraus resultierenden Möglichkeiten an Touch-Gesten liegt der Fokus dieser Arbeit auf drei Szenarien: *Fahren*, *Drehen* und einen Gegenstand *Aufnehmen* bzw. *Ablegen*. Diese Szenarien wurden ausgewählt, da sie sich mit Hilfe der Basis-Touch-Gesten [Vi10] umsetzen lassen und sich an Grundfähigkeiten mobiler, nicht-humanoider Roboter orientieren. In einem zweistufigen Nutzertest wurde insbesondere die Intuitivität der Touch-Gesten zunächst ohne und später mit Vorgaben zur Ausführung der Touch-Gesten untersucht.

Um diese dynamischen Bewegungen der Touch-Gesten im Nachhinein wieder sichtbar zu machen und exakt auswerten zu können, wurde eine Methodik zur Evaluation entwickelt. Diese Methodik beinhaltet Versuchsaufbau, -durchführung und -auswertung. Ziel des Tests für die Nutzer war es, eine Teststrecke zu absolvieren, welche die ausgewählten Szenarien beinhaltete. Dafür befand sich der Modellroboter zentral zwischen einem aufzunehmenden Objekt und einer Zielfläche, wobei die Ausrichtung des Modellroboters etwa 45 ° nach rechts zum Objekt gedreht war. Die Aufgabe war es, dass der Modellroboter das Objekt aufnimmt und an der Zielfläche ablegt. Dafür wurden den Nutzern zunächst keinerlei Vorgaben zur Ausführung oder zur Lösung der Aufgabe gegeben. Vor Beginn des Tests erhielten die Nutzer Hinweise zu den Fähigkeiten des Modellroboters. Der zweite Durchgang des Nutzertests unterscheidet sich vor allem durch konkretere Vorgaben zur Ausführung der Touch-Gesten. Außerdem kamen mehrere Geräte unterschiedlicher Formfaktoren zum Einsatz, um herauszufinden, ob und gegebenenfalls wie die Größe des Geräts die Eingabe der Touch-Gesten beeinflusst.

Für die Durchführung des Nutzertests absolvierten die Teilnehmer die vorgegebene Teststrecke unabhängig voneinander. Um im Gegensatz zur weitverbreiteten Videoauswertung eine effizientere und präzisere Möglichkeit der Auswertung von Touch-Gesten zu bieten, wurden zwei Anwendungen entwickelt. Zunächst wird durch eine Applikation auf dem Android-Eingabegerät die Speicherung der eingegebenen Touch-Gesten zur Laufzeit ermöglicht. Die Eingaben werden mit exakten Werte für Zeitangaben und Koordinaten der Finger als Log-Dateien auf dem Gerät abgespeichert. Anschließend können diese Dateien mit Hilfe eines Web-Tools automatisiert ausgelesen und die Touch-Gesten-Eingaben exakt visualisiert und präzise miteinander verglichen werden (Abb. 1). Dieses System trägt mit seiner Versuchsdurchführung und -auswertung erheblich zur Vereinfachung und Effizienzsteigerung der Evaluation bei.

Die Darstellung der Berührungspunkte, welche die gespeicherten Koordinaten der Finger auf dem Display sind, werden mit zunehmender Zeit kleiner, sodass Rückschlusse auf die Eingaberichtung der Touch-Geste gezogen werden können. In Abbildung 1 sind die jeweiligen Eingaberichtungen mit Hilfe von Pfeilen verdeutlicht. Weiterhin kann anhand der Visualisierung der Touch-Gesten abgelesen werden, wie schnell oder langsam die Eingabe verlaufen ist: Je kleiner der Abstand zwischen zwei Punkten ist, desto langsamer war die Eingabe; je größer der Abstand ist, desto schneller die Eingabe. Um die Touch-Gesten im weiteren Verlauf farblich voneinander unterschieden zu können, werden die Drag-Touch-Gesten (Wischen) für die Aktion *Fahren* blau dargestellt, die Rotate-Touch-Gesten (Rotieren) zum *Drehen* rot visualisiert und die Pinch- bzw. Spread-Touch-Geste (Kneifen bzw. Spreitzen) zum *Aufnehmen* und *Ablegen* durch eine grüne Färbung kenntliche gemacht.

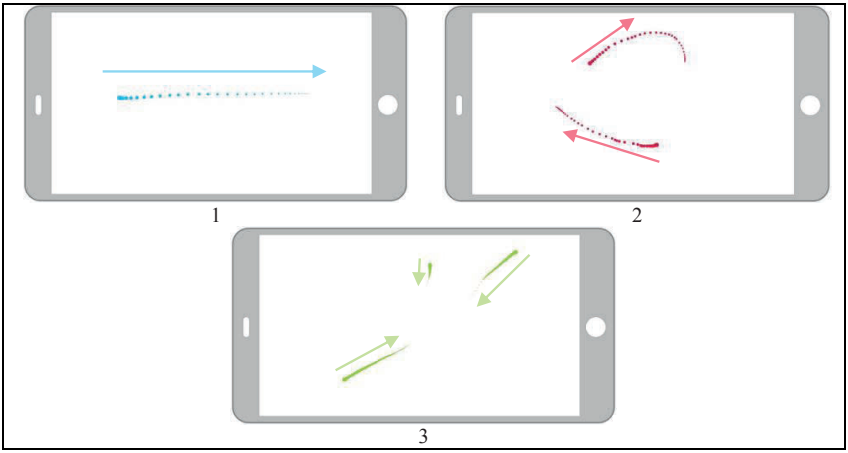


Abbildung 1: Verdeutlichung der Eingaberichtung und Farbmuster zur Unterscheidung der Touch-Gesten. Blau: Drag-Touch-Geste. Rot: Rotate-Touch-Geste. Grün: Pinch- bzw. Spread-Touch-Geste.

3 Evaluation

Zur Bewertung der Steuerung eines Roboters durch die Eingabe von Touch-Gesten werden üblicherweise Usability-Tests durchgeführt. Sie geben Aufschluss über die allgemeine Bedienbarkeit von Benutzerschnittstellen. Um den in dieser Arbeit verfolgten Ansatz einer Robotersteuerung mithilfe von Touch-Gesten zu evaluieren, wurde zunächst der grundsätzliche Umgang der Nutzer mit dieser Art von „Fernsteuerung“ festgestellt. Dabei stand nicht nur die Frage im Fokus, welche Touch-Gesten generell verwendet wurden, sondern auch wie die Nutzer mit einem Modellroboter und dem einfachen Design der Schnittstelle interagierten.

An diesem Nutzertest nahmen insgesamt 15 Probanden teil. Die Acht männlichen und sieben weiblichen Teilnehmer waren im Alter von 20 bis 40 Jahren. Alle Testpersonen haben täglichen Umgang mit Smartphones oder Tablets und sind daher mit (Multi-) Touch-Gesten vertraut. Die Tests wurden unabhängig voneinander durchgeführt.

Die Teilnehmer erhielten zur Durchführung der ersten Stufe des Nutzertests zur Einschätzung der Intuitivität einer Touch-Gesten-Steuerung ein 5 Zoll Smartphone, da die Größe des Eingabegeräts zu diesem Zeitpunkt der Auswertung noch nicht relevant war. Später absolvierten die Nutzer den Test mit Eingabegeräten unterschiedlicher Formate (5 Zoll Smartphone bis 12 Zoll Tablet). In der ersten Stufe des Nutzertests kam ein Modellroboter aus Papier zum Einsatz. In der zweiten Stufen wurde dieser durch einen Modellroboter aus Lego ersetzt, der aufgrund seiner Beschaffenheit weniger Interpretationsspielraum zu seiner Funktionalität zuließ. Um die verwendeten Touch-Gesten der Nutzer im Nachhinein sichtbar zu machen und damit auswerten und vergleichen zu können, wurden während der Lösung der Aufgaben die Fingerbewegungen der Nutzer aufgezeichnet. Zudem wurden die Probanden während der Untersuchung dazu aufgefordert, laut zu denken, um ihre Erwartungen und Intentionen nachzuverfolgen sowie gegebenenfalls Rückfragen zur Vorgehensweise zu stellen.

Hierfür beobachtete der Moderator des Nutzertests während der Durchführung zum einen die Nutzer und zum anderen übernahm er die Aufgabe, den Modellroboter nach Eingabe der jeweiligen Touch-Gesten erwartungskonform zu bewegen. Nur so konnte die fehlerfreie Ausführung der vom Nutzer intendierten Bewegungen garantiert werden. Der Moderator simulierte zudem mögliche Probleme, auf die der Nutzer reagieren sollte. Die Lösung dieser Probleme durch den Nutzer kann ebenfalls im Nachgang in die Auswertung einbezogen werden.

Fahren

Eine Teilaufgabe des Nutzertests war das Bewegen des Roboters von einer Position zu einer anderen. Ohne Vorgaben zur Ausführung der Touch-Gesten erwies sich dieser zunächst als einfach erscheinende Anwendungsfall als komplexe Problematik. Entscheidend für die Wahl einer Touch-Geste waren die Vorkenntnisse der Nutzer. Zum Teil unterschieden sich die Touch-Gesten zur Bewältigung dieser Aufgabe stark.

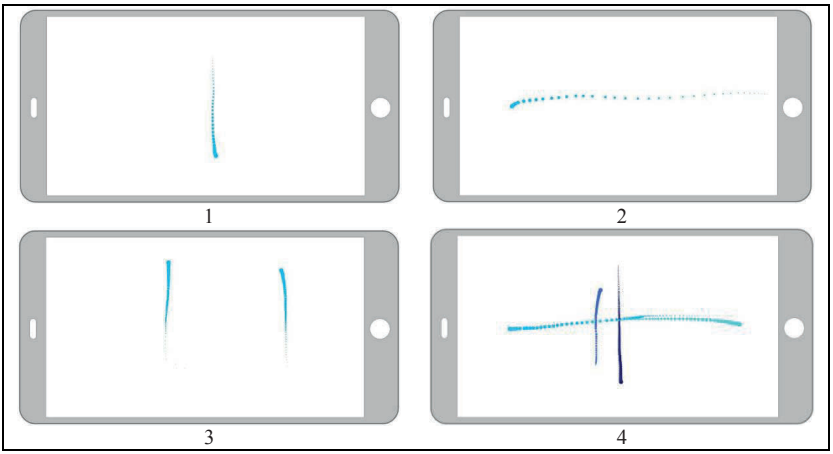


Abbildung 2: Visualisierung der verschieden ausgeführten Touch-Geste zur Aktion *Fahren*. 1: Drag-Geste nach oben. 2: Drag-Geste nach rechts. 3: Drag-Geste nach unten mit zwei parallel geführten Fingern. 4: Drag-Gesten eines Nutzers in alle Richtungen.

In Abbildung 2 sind deutlich voneinander abweichende Touch-Eingaben für die Aktion *Fahren* dargestellt. Grundsätzlich basiert die Mehrheit der Eingaben auf dem Konzept der Drag-Geste (Wischen) und somit auf dem Vorwissen der Nutzer. Damit in Zusammenhang steht auch die Verwendung von ausschließlich einem Finger zur Ausführung; in lediglich zwei Fällen wurden zwei Finger verwendet (Abb. 2.3). Ansonsten war der variierende Faktor jedoch die Richtung, in welche die Touch-Gesten ausgeführt wurden. Entscheidend dafür waren der Erfahrungsstand der Nutzer und das Aussehen des Roboters, denn diese beiden Umstände beeinflussten die Wahl der Touch-Gesten.

Während sich in den Abbildungen 2.1 - 2.3 die Eingaberichtungen verschiedener Nutzer voneinander unterscheiden, aber pro einzelnem Nutzer einheitlich sind, verwendeten andere Nutzer während der Lösung der Aufgabe wiederholt unterschiedliche Eingaberichtungen (Abb. 2.4), je nach Ausrichtung des Roboters zum Nutzer. Dass die Ausrichtung des Roboters eine so maßgebliche Rolle für die Eingaberichtung der Touch-Geste spielt, ist eine interessante Beobachtung. Diese Herangehensweise wurde von mehreren Nutzern intuitiv umgesetzt. In Bezug auf die Ausrichtung des Roboters ist Abbildung 3 ein bezeichnendes Beispiel. Sie repräsentiert die Position des Roboters in einer Art Draufsicht. Anhand der Visualisierung der Touch-Gesten dieses Nutzers ist die Bewegung des Roboters gut nachvollziehbar. Zum Vergleich ist die Musterlösung zusätzlich abgebildet. Interessant ist auch die verwendete Touch-Geste in Abbildung 2.3: Hierin spiegelt sich der Einfluss des Erscheinungsbildes des Modellroboters wider, welcher an ein Kettenfahrzeug erinnerte. Durch die Ausführung der Touch-Geste mit zwei Fingern wollten die entsprechenden Testteilnehmer beide Ketten ansteuern.

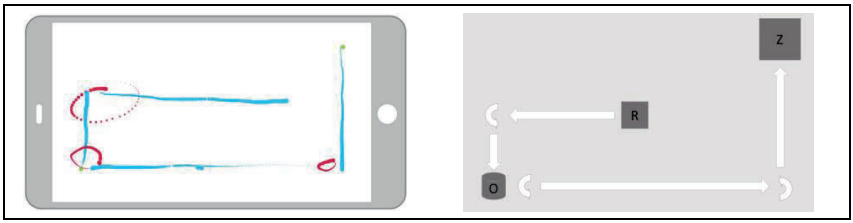


Abbildung 3: Räumliche Abbildung des Roboters auf dem Display. Links: Anhand der Visualisierung der eingegebenen Touch-Gesten ist der Aufbau des Tests erkennbar. Rechts: Zum Vergleich schematischer Aufbau des Nutzertests. In der Ausgangssituation befindet sich der Roboter (R) zentral zwischen dem Objekt (O) und der Zielfläche (Z).

Drehen

Der zweite Teilaspekt des Nutzertests war das *Drehen* des Roboters, um die Fahrtrichtung zu ändern. Zwar gab es weniger grundlegende Schwierigkeiten bei der Ausführung, dennoch ist die Rotations-Geste an sich eine komplexere Touch-Geste [Hi11], sodass die Ausführungen stark variierten. Zunächst unterscheidet sich die Anzahl der verwendeten Finger (einer oder zwei): In den Abbildungen 4.1 und 4.2 ist zu erkennen, dass jeweils nur ein Finger zur Eingabe der Touch-Geste verwendet wurde. Einige Nutzer empfanden die Verwendung eines einzigen Fingers intuitiver, insbesondere nach Eingabe der ebenfalls mit einem Finger ausgeführten Wisch-Geste. Wesentliche Unterschiede in der Ausführung der Rotations-Geste mit einem Finger ist die Größe des Kreisbogens: Nahezu alle Variationen zwischen einem Halbkreis (Abb. 4.2) und einem vollständigen Kreis (Abb. 4.1) waren während dieses Nutzertests vertreten, da es keine exakten Vorgaben zum Drehwinkel des Roboters gab und die Nutzer zudem durch das Kommentieren ihrer Eingabe verdeutlichten, wie weit sich der Roboter drehen sollte.

In den Abbildungen 4.3 - 4.5 hingegen setzten die Nutzer die Touch-Gesten mit zwei Fingern um. Aber auch hier lassen sich offensichtliche Unterschiede erkennen. Einerseits war festzustellen, dass der Ansatz verfolgt wurde, bei dem sich ein Finger kaum bewegt und damit einem Fixpunkt gleicht, um welchen sich der andere dreht. Andererseits können sich beide Finger gegeneinander bewegen und damit eine Kreisbewegung ausführen. Einen Spezialfall stellt die Touch-Geste dar, bei der ein vollständiger Kreis mit zwei parallel geführten Fingern eingegeben wurde (Abb. 4.5).

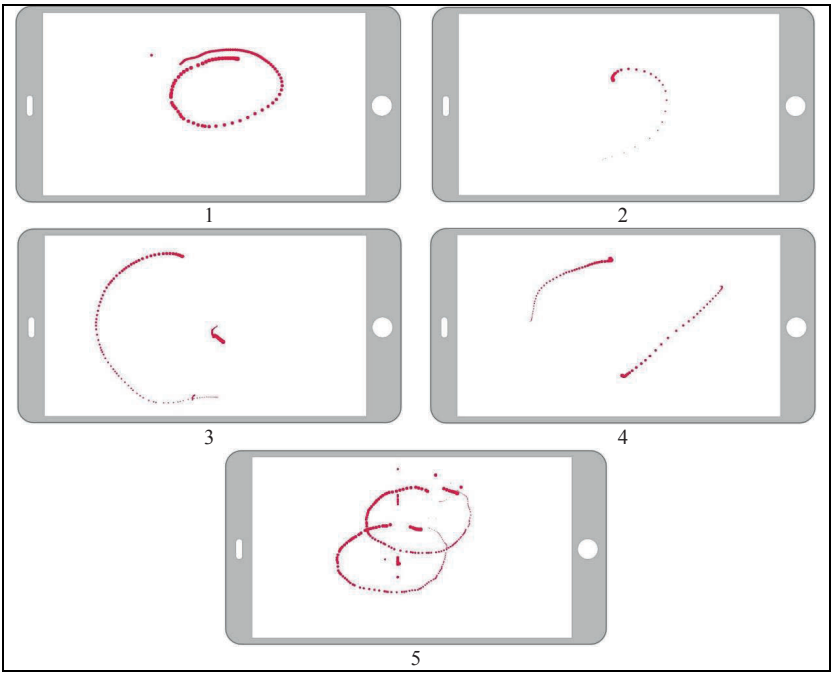


Abbildung 4: Visualisierung der verschieden ausgeführten Touch-Gesten zur Aktion *Drehen*. 1: vollständiger Kreis mit einem Finger. 2: Halbkreis mit einem Finger. 3: Nutzung von zwei Fingern, wobei einer als Fixpunkt dient. 4: Zwei Fingern bewegen sich gegeneinander. 5: Vollständiger Kreis mit zwei parallel geführten Fingern.

Objekt aufnehmen und ablegen

Der letzte Schwerpunkt der zu lösenden Aufgabe war das *Aufnehmen* und spätere *Ablegen* eines Objektes. Bei diesem Szenario war auffällig, dass im Gegensatz zu den ersten beiden Teilaufgaben mehrheitlich die gleiche Touch-Geste verwendet bzw. der gleiche Ansatz verfolgt wurde. Für diese Szenarien nutzten die Testpersonen am häufigsten die Pinch-Touch-Geste, um einen Gegenstand aufzunehmen, und die Spread-Touch-Geste, um einen Gegenstand wieder abzulegen (siehe Abb. 5.1 und 5.2). Führten die Nutzer komplexere Gesten aus, lagen dennoch diese beiden Gesten zugrunde (Abb. 5.3 und 5.4).

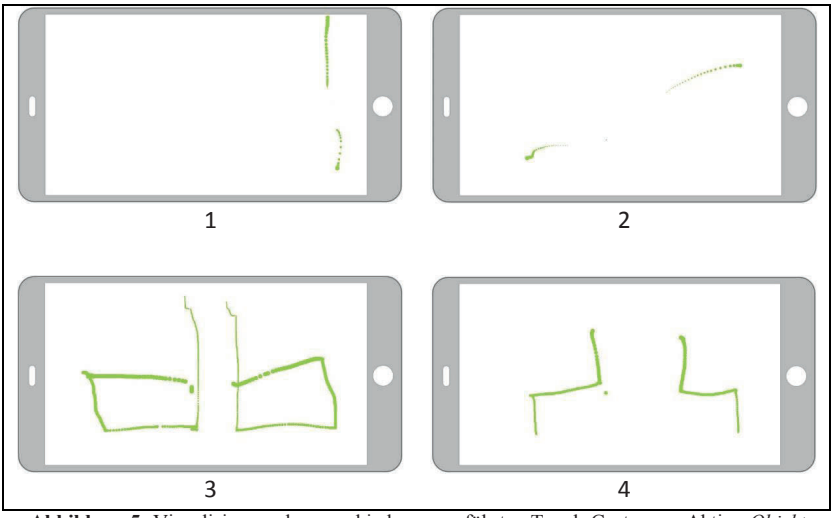


Abbildung 5: Visualisierung der verschieden ausgeführten Touch-Gesten zur Aktion *Objekt aufnehmen/ablegen*. 1: Vertikal ausgeführte Pinch-Touch-Geste. 2: Diagonal ausgeführte Pinch-Touch-Geste. 3: Komplexe Touch-Geste zum Aufnehmen eines Objekts. 4: Komplexe Touch-Geste zum Ablegen eines Objekts.

Hintergrund der Verwendung dieser Touch-Geste für diesen Anwendungsfall scheint die Übertragung der natürlichen Greifbewegung auf eine abstraktere Ebene zu sein. Daher eignen sich die Pinch- und Spread-Touch-Geste hervorragend. Unterschiede in der Ausführung liegen vor allem im Eingabebereich der Touch-Geste (Abb. 5.1 und 5.2) und der Anzahl der verwendeten Finger (zwei oder drei). Je nach Auffassung, wie genau dem Roboter die Aufgabe vermittelt werden muss, wurden die Eingaben wie in Abbildung 5.3 und 5.4 komplexer: Bei Ausführung dieser Touch-Gesten wurde nicht nur das *Aufnehmen* bzw. *Ablegen* berücksichtigt, sondern zusätzlich das Heben und Senken der Arme einbezogen. Außerdem konnten für die Informationsübermittlung zwei verschiedene Herangehensweisen beobachtet werden. Zum einen wurden komplexe Touch-Gesten wie in Abbildung 5.3 und 5.4 eingegeben, die voraussetzen, dass der Roboter die Aktionen bereits während ihrer Eingabe ausführt. Andererseits wurden einzelne Touch-Gesten eingegeben, auf die der Roboter nach jeweiliger Beendigung der Eingabe reagiert.

Auswertung der Formfaktoren

Die zweite Stufe des Nutzertests basiert auf den Erkenntnissen des ersten Durchgangs. Da die verwendeten Touch-Gesten aus dem ersten Nutzertest stark variierten, bekamen die Teilnehmer konkrete Vorgaben zur Ausführung der Touch-Gesten. Dadurch sollte der Einfluss unterschiedlicher Größen der Eingabegeräte (5 Zoll, 8 Zoll und 12 Zoll Gerät) auf die Ausführung der Touch-Gesten nachvollziehbarer evaluiert werden. Die Ausführung der Touch-Gesten sollte nach folgenden Kriterien ablaufen: Zum *Fahren* des Modellroboters sollte eine Drag-Touch-Geste mit einem Finger in eine einheitliche Richtung verwendet werden, zum *Drehen* eine Rotations-Touch-Geste mit zwei Fingern und zum *Aufnehmen* bzw. zum *Ablegen* eines Objekts eine Pinch- bzw. Spread-Touch-Geste mit drei Fingern. Für die Aktionen *Fahren* und *Drehen* stellten die eben beschriebenen Gesten Start-Gesten dar, um die entsprechende Aktion zu beginnen. Um diese Aktionen zu beenden, sollte eine Abbruch-Geste eingegeben werden, damit der Nutzer die Distanz, die der Modellroboter zurücklegen soll, und den Winkel, um den sich der Modellroboter drehen soll, beeinflussen konnte. Dabei sollten die Nutzer dieselbe Teststrecke wie im ersten Durchgang unter Anwendung der vorgegebenen Touch-Gesten auf allen Geräten lösen. Die Reihenfolge, in welcher die unterschiedlichen Geräte verwendet wurden, war randomisiert.

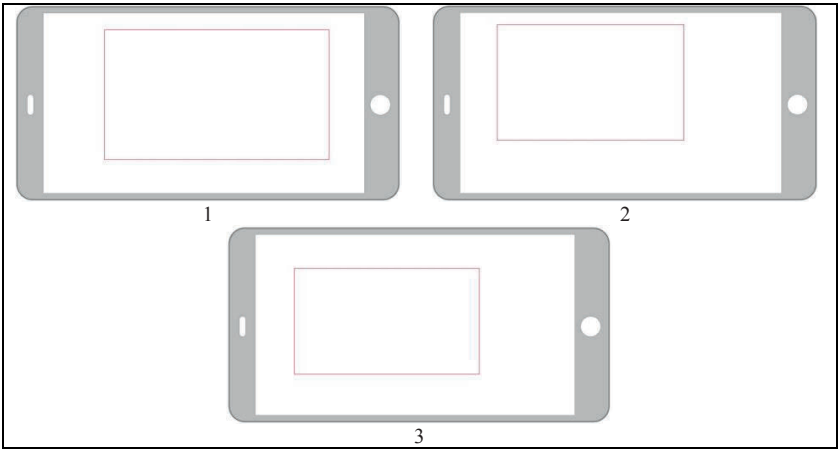


Abbildung 6: Visualisierung der Eingabegeräte der Rotate-Touch-Geste bei unterschiedlichen Formfaktoren. 1: 5 Zoll Gerät. 2: 8 Zoll Gerät. 3: 12 Zoll Gerät

Im Nachhinein konnten verschiedene Aspekte der Touch-Gesten-Eingabe aufgrund der Formfaktoren der Eingabegeräte ausgewertet werden: Größe, Dauer und Position der Touch-Gesten. So fiel bei der Größe der Touch-Gesten auf, dass sich die tatsächliche Ausdehnung (in Pixeln) der Eingaben weniger unterschied als die Ausdehnung der relativ zur Displaygröße ausgeführten Eingaben. Analog dazu wurden auf dem 5 Zoll Gerät größere Bereiche des Displays zur Ausführung der Touch-Gesten verwendet als auf den beiden größeren Eingabegeräten (siehe Tabelle 1). Dennoch ergaben sich auf allen Geräten ungenutzte Bereiche, sodass eventuelle Schaltflächen dort platziert werden können, ohne den Nutzer in seinem Eingabeverhalten zu beeinflussen (Abb. 6). Bei der Dauer der Eingabe der Touch-Gesten konnte eine Relation zur Größe und Komplexität (wie viele Finger für die Eingabe verwendet wurden) festgestellt werden.

Tabelle 1: Position der Rotate-Touch-Geste auf verschiedenen Eingabegeräten. Die Ergebnisse lassen deutlich die geringe Ausnutzung des verfügbaren Eingabebereichs erkennen.

		Smartphone (5 Zoll)		**Tablet (8 Zoll)**		**Tablet (12 Zoll)**	
		px	**%**	**px**	**%**	**px**	**%**
Position	**Min. Position x (Horizontale)**	342	19,1	221	11,5	311	12,1
	Max. Position x (Horizontale)	1597	89,0	1338	69,7	1795	70,1
	Diff. Position x (Horizontale)	1255	70,0	1117	58,2	1484	58,0
	Min. Position y (Vertikale)	100	9,3	73	6,6	299	18,7
	Max. Position y (Vertikale)	878	81,3	779	70,6	1237	77,3
	Diff. Position y (Vertikale)	778	72,0	706	64,0	938	58,6
Genutzter Eingabebereich		50 %		37 %		34 %	

4 Diskussion

Dieser Nutzertest hat bereits einige grundsätzliche Schwierigkeiten einer Touch-Gesten-Steuerung aufgezeigt, aber auch, dass diese Art der Steuerung durchaus geeignet und intuitiv ist. Da es sich bei dieser Steuerungsmöglichkeit um eine „natürliche" Interaktionsform handelt, ist die Anzahl der Eingabemöglichkeiten für Nutzer ohne Vorgaben zu diesen nicht ersichtlich. Doch obwohl nicht zuletzt diese Tatsache zu stark variierenden Ansätzen der Touch-Gesten-Eingabe für gleiche Aktionen führte, fiel es den Nutzern nicht schwer, für sie intuitive Möglichkeiten der Ausführung zu finden. Kennen Nutzer die konkreten Interaktionsmöglichkeiten nicht, basiert ihr Verhalten auf ihrem Wissen über die (Basis-)Touch-Gesten und Annahmen über den Roboter und dessen Verhalten sowie auf ihren Erfahrungen mit Fernsteuerungen. Insbesondere letzteres ist in diesem Zusammenhang ein wichtiger Einflussfaktor für die Bedienung einer Touch-Gesten-Steuerung: Sind Nutzer bereits mit anderen Steuerungsmöglichkeiten bspw. aus Computerspielen oder ferngesteuerten Fahrzeugen vertraut, können sie dieses erworbene Wissen auf eine Steuerung mittels Touch-Gesten übertragen und haben daher weniger Probleme mit der Position und Ausrichtung des Roboters zum Nutzer.

Ein weiterer beeinflussender Faktor für das Eingabeverhalten der Nutzer ist das Aussehen des Roboters und die ihm damit vom Nutzer zugeschriebenen Eigenschaften. Auffällig waren die Unterschiede der Eingaben zwischen den beiden eingesetzten Modellrobotern zunächst aus Papier und später aus Lego. Wesentlich weniger Probleme bereitete der Lego-Modellroboter, da seine Funktionen besser ersichtlich waren als beim Papier-Modellroboter. Aus diesem Grund wählten einige Nutzer, insbesondere im ersten Durchgang des Tests, mehrere oder komplexere Touch-Gesten, um eine Teilaufgabe zu lösen, obwohl die Eingabe einer einfachen Touch-Geste ausreichend gewesen wäre, da der Roboter teilweise autonom reagiert.

Außerdem fielen bei der Lösung der Aufgabe zwei Herangehensweisen auf: Auf der einen Seite erwarteten einige Nutzer eine Reaktion des Roboters auf die Touch-Gesten erst nach ihrer abgeschlossenen Eingabe. Auf der anderen Seite gab es Nutzer, die bereits während der Ausführung der Touch-Gesten auf die entsprechende Aktion des Roboters warteten, um bestimmte Aktionen, wie *Fahren* oder *Drehen*, direkt beeinflussen zu können. Damit würde eine Aktion abbrechen, sobald die Touch-Gesten-Eingabe beendet wurde. Diese Problematik des direkten Einflussnehmens auf die Dauer der Ausführung einer Aktion kann durch die Eingabe einer Start- sowie einer Stopp-Touch-Geste entgegnet werden. Insbesondere bei den Aktionen *Fahren* und *Drehen* ist es sinnvoll, dem Roboter Anweisungen zu geben, wann die Aktion beendet werden soll, um zu vermeiden, dass er sich unbestimmt im Kreis dreht oder geradeaus fährt. Die aus dem Nutzertest hervorgegangenen Faktoren beeinflussen und erschweren eine endgültige Realisierung einer Touch-Gesten-Steuerung. Die entwickelte Methode ermöglicht allerdings, dass diese und andere Aspekte in weiteren Untersuchungen evaluiert werden können.

Literaturverzeichnis

[Hi11] Hinrichs, U.: Carpendale, S.: Gestures in The Wild: Studying Multi-Touch Gestures Sequences on Interactive Tabletop Exhibits. Conference on Human Factors in Computing Systems, Vancouver, 2011; S. 3023-3032.

[Na13] Nacenta, M. A.; Kamber, Y.; Qiang, Y.; Kristensson, P. O.: Memorability of Pre-designed & User-defined Gesture Sets. Conference on Human Factors in Computing Systems, Paris, 2013; S. 1099-1108.

[No02] Norman, D.A.: The Design of Everday Things. Basic Books, New York, 2002.

[Ol14] Olafsdottir, H.; Appert, C.: Multi-Touch Gestures for Discrete and Continous Control. International Working Conference on Advanced Visual Interface, Como, 2014; S. 177-184.

[Vi10] Villamor, C.; Willis, D.; Wroblewski, L.: Touch Gesture Reference Guide. In: http://www.lukew.com/ff/entry.asp?1071 (13.03.2016).

Entwurf und Implementierung von Optimierungs- und Funktionserweiterungen der hexagonalen Bildrasterung in der Videokompressionssoftware x264HMod

Tobias Schlosser, Robert Manthey*, Marc Ritter*

Junior Professorship Media Computing
Technische Universität Chemnitz
09107 Chemnitz, Germany
tobias.schlosser@informatik.tu-chemnitz.de

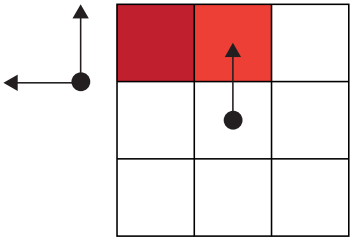
Abstract: Auf Grund der fortschreitenden Digitalisierung und Digitalität von Inhalten im Bereich von Bild und Video, lässt sich der Verarbeitung und Speicherung jener Daten ein zunehmend höherer Stellenwert zuordnen. Hier kommen in der Praxis nach aktuellem Stand der Technik von Aufnahme- und Ausgabegeräten fast ausschließlich viereckige Rasterformate zum Einsatz. In meiner Bachelorarbeit habe ich dazu einen alternativen Ansatz verfolgt, welcher nach biologischem Vorbild die Verwendung von hexagonalen Rastern bzw. Bildpunkten vorsieht. Deren entscheidende Vorteile, wie beispielsweise eine erhöhte Abbildungseffizienz, wurden mit Hilfe des in Schlosser [Sch16] vorgestellten Hexagonal Image Processing Frameworks CHIP, welches die Funktionalität der Transformation bietet, untersucht und entsprechend dargestellt. Das Ziel bestand darin, dieses zu optimieren und um verschiedene Verfahren zur Bildverarbeitung zu erweitern. Neben dem Einbau in den Encoder x264 zur Videoverarbeitung, erfolgte zudem die Entwicklung eines Wiedergabegerätes, wodurch die hexagonalen Pixel adäquat visualisiert werden können. Die Ergebnisse zeigen, dass sich nicht nur eine Echtzeitfähigkeit erzielen lässt, sondern auch ein qualitativ überzeugendes Transformationsergebnis.

Keywords: Image and Video Processing, Digital Signal Processing, Hexagonal Grid

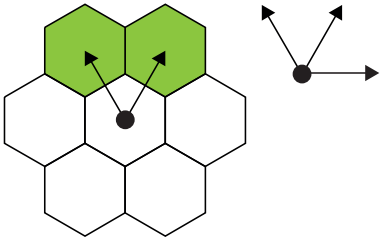
1 Motivation

Um reale Bilder mit Computern speichern und verarbeiten zu können, sollten diese in einem digitalisierten Format vorliegen bzw. so beschaffen sein, dass diese rechnerisch zu erfassen und sämtliche wichtigen Informationen aus ihnen zu entnehmen sind. Bei diesen Informationen handelt es sich vor allem um die Eigenschaften wie Farbe, Helligkeit und Größe, also die Menge der darzustellenden Daten des Bildes. So entscheidet beispielsweise der Prozess, welcher aus einem realen Bild ein durch den Computer lesbares Format erzeugt (auch Digitalisierung genannt), über die Organisation und die verwendete Art oder Technik bei der Bildaufnahme, über die Struktur der Daten. [Sch16, S. 1]

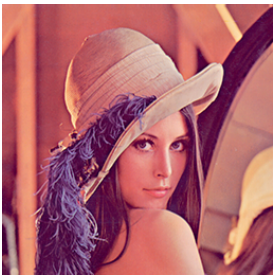
*Wissenschaftlicher Betreuer



(a) Quadratisches Raster

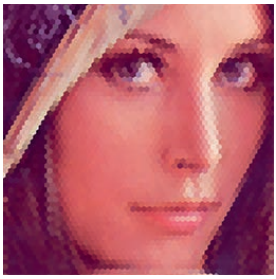


(b) Hexagonales Raster



(c) Originalbild

(d) Ausschnitt auf quadratischem Raster

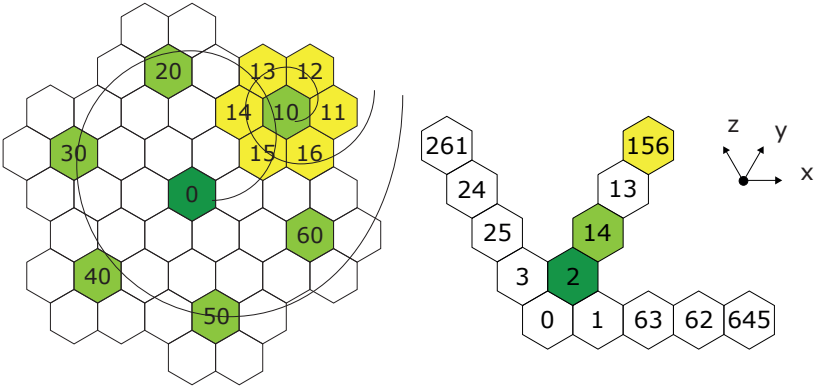


(e) Ausschnitt auf hexagonalem Raster

Abbildung 1: Gegenüberstellung Rasterformate [Sch16, S. 2] (Testbild „Lena“ [Web16])

Bei den sogenannten Pixelgrafiken handelt es sich um eine solche Art der Organisation, wobei die entsprechenden Bildpunkte dieser (auch Pixel) mit jedem Farbwert einzeln angegeben werden. Hier kommen in der Praxis aufgrund des aktuellen Standes der Technik fast ausschließlich viereckige bzw. quadratische Bildpunkte zum Einsatz. Einen weiteren Ansatz stellt hingegen die Verwendung von hexagonalen Pixeln dar, welche ihre entscheidenden Vorteile im Besonderen durch die biologische Evolution von Augen manifestieren. So weist unter anderem die menschliche Retina eine stark ausgeprägte hexagonale Anordnung der Sehnerven auf, wodurch die anschließende Verarbeitung von auftreffenden Signalen deutlich effizienter erfolgen kann (nach [MS05, S. 6 ff.]).

Abbildung 1 zeigt einen Vergleich der beiden Rasterformate (a und b) mit jeweils einem Ausschnitt für das Testbild Lena darunter (d und e). Diese lassen bereits erste Vorteile des hexagonalen Rasters, wie beispielsweise homogene Pixelabstände und Trennflächen zu benachbarten Bildpunkten, erkennen, woraus fernerhin die Eindeutigkeit der Nachbarschaft folgt. Zusammen mit der um ca. 12 % erhöhten Abbildungseffizienz (nach [Nel89]), welche eine Speicherung von größeren Mengen an Informationen auf der gleichen Anzahl Pixel ermöglicht, resultieren zudem geringere Rechenzeiten und eine erhöhte Effizienz beim Programmieren (nach [Gol69]).



(a) Aufbau Spiralarchitektur

(b) Optimierung Multiplikation Hexints

Abbildung 2: Hexagonales Raster in CHIP [Sch16, S. 10 ff.]

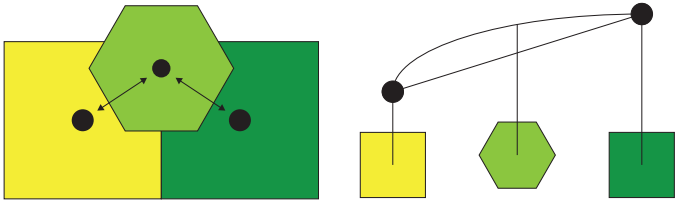
Auf Grund der zu großen Teilen fehlenden theoretischen und praktischen Umsetzung dieses, war es das Ziel der Bachelorarbeit, dass sogenannte Hexagonal Image Processing Framework CHIP, welches zuvor eigens aus HIP [MS05] entwickelt wurde, an geeigneten Stellen zu erweitern und zu optimieren. Dieses bietet die Hin- und Rücktransformation auf Basis eines gegebenen Viereck-Rasterformates in das obig dargestellte Hexagonale. Weiterhin existieren verschiedene Verfahren zur Einzelbildverarbeitung wie DCT, Quantisierung, Deblocking- und Kantendetektions-Filter.

Da das CHIP-Framework nur auf Einzelbildern arbeiten kann, sollte des Weiteren der Einbau dieses zusammen mit weiteren Verfahren in den Encoder x264 zur Videoverarbeitung erfolgen. Dessen Erweiterung (genannt x264HMod) sieht eine Umwandlung der eingelesenen Einzelbilder (auch Frames) in das hexagonale Rasterformat, anschließende Bearbeitung durch CHIP und letztendlich wieder die Rücktransformation sowie Ausgabe in Form eines Videos, vor.

Im Rahmen dieser Arbeit erfolgt eine Übersicht der entscheidenden Implementierungs- und Optimierungsschritte, die in [Sch16] und den nachfolgenden Wochen realisiert wurden.

2 Analyse des Optimierungspotentials der hexagonalen Bildrasterung

Da sich hexagonale Bildpunkte nur schlecht über ein Zeilen- und Spaltenformat adressieren lassen, kommt im CHIP-Framework die Methode des sogenannten *Spiral Structure Adressings* (nach [MS05, S. 41]; engl. für strukturelle Spiraladressierung) zum Einsatz. Abbildung 2 (a) zeigt diese am Beispiel des Sechseck-Rasters, auch Hexarray genannt,



(a) Interpolation eines hexagonalen Pixels

(b) Vergleich bilineare und bikubische Interpolation

Abbildung 3: Bilineare und bikubische hexagonale Interpolation [Sch16, S. 11]

der Größe 49. Der Aufbau eines solchen erfolgt entlang der dargestellten Nummerierung, wobei die einzelnen Bildpunkte, aus welchen es konstruiert wird (genannt Hexints), in Abhängigkeit von ihrer aktuellen Position im Hexarray, immer aus den Ziffern 0 bis 6 bestehen. Entsprechend dieser Vorschrift bilden sieben Sub-Hexarrays der Größe 7^{n-1} eines der Größe 7^n und sieben Hexints (7^1) das Kleinste (Ordnung n bis 1).

Obwohl sich diese Art der Adressierung im Vergleich zu seinem viereckigen Pendant umständlicher und weniger intuitiv darstellt, birgt sie jedoch auch einen entscheidenden Vorteil. Soll beispielsweise der Hexint mit der Nummer 16 bestimmt werden, ist es möglich, diesen über die Positionen der Hexints 10 und 6 zu lokalisieren. Dies ist der Fall, da 16 die gleiche Position wie 6 besitzt, wenn 10 als der Koordinatenursprung angenommen wird. Somit reicht eine einfache Addition der Koordinaten beider Hexints für eine Lokalisierung von 16 aus. [Sch16, S. 10]

Zur Erzeugung der eigentlichen Farbwerte kommen in CHIP auf Basis dieser Struktur die beiden Methoden der bilinearen und bikubischen Interpolation zum Einsatz. Bei diesen handelt es sich um Verfahren zur Zwischenwertbestimmung, welche über das originale Raster versuchen, die Farbwerte der neuen hexagonalen Bildpunkte approximiert zu bestimmen (Beispiel siehe Abbildung 3 (a)). Die bilineare Interpolation setzt dazu sämtliche quadratischen Bildpunkte innerhalb eines Radiusses in ein lineares Verhältnis, wobei die bikubische dies hingegen mittels kubischer bzw. polynomieller Funktionalität bewerkstelligt (b).

Um die Performanz sowie Qualität der Transformation selbst optimieren zu können, wurden folgende Ansatzpunkte für Verbesserungen identifiziert:

- Vorberechnungen: Bei der Transformation kann es aufgrund des Radiusses der Interpolation dazu kommen, dass Eigenschaften von Hexints, wie beispielsweise Koordinaten, mehrfach berechnet werden (überlappende Radien). Diese Eigenschaften lassen sich entsprechend vorab ermitteln und zum Start des Frameworks einspeisen.
- Abstandsberechnungen: Für die Bestimmung von Abständen kommt in HIP unter anderem eine Art Multiplikation zum Einsatz, welche einer n-fachen Addition ei-

nes Hexints mit sich selbst entspricht. Dabei kann es auch geschehen, dass Teilsummen mehrfach berechnet werden. Abbildung 2 (b) zeigt ein Beispiel mit dem Hexint 2 (Koordinate $[0, 1, 0]$), welcher drei Mal mit sich selbst zu verrechnen ist. Hier könnten die Zwischenergebnisse aus jeweils $(2 + 2)$ so zusammengefasst werden, dass nur noch die Vielfachen von 2 zu bestimmen wären. Somit genügen mit $(2 + 2) = 14$ ($[0, 2, 0]$) und $(14 + 14) =$ Hexint 156 ($[0, 4, 0]$) zwei, anstatt der ursprünglichen drei Additionen zur Lösung des Problems.

- Interpolationsnormalisierung: Dadurch, dass die zum Einsatz kommenden Interpolationsmethoden mehrere Bildpunkte zur Zwischenwertbestimmung betrachten, müssen die Ergebnisse aus den einzelnen Berechnungen zusätzlich noch normalisiert werden. Dieser Schritt entfällt jedoch in HIP, was wiederum zur Folge hat, dass ein zeilenweise zyklischer Fehler auftritt, welcher die Bildqualität deutlich mindern kann (Beispiel siehe Abbildung 7 (b)).
- Parallelisierung: Da die Transformation zeilenweise arbeitet, ist es weiterhin möglich, die Berechnungen einer jeden Zeile jeweils einem Thread zuzuweisen. Hier wäre eine Beschleunigung bis annähernd Faktor #*Zeilen* denkbar.

3 Implementierung

Dieses Kapitel befasst sich mit der Erweiterung und Optimierung des CHIP-Frameworks sowie dem Einbau einiger ausgewählter Verfahren zur Einzelbild- und Videoverarbeitung aus Kapitel 1. Weiterhin erfolgt die Erläuterung der Integration dieser in x264 (x264HMod).

3.1 Hexagonal Image Processing Framework CHIP

In dem der Bachelorarbeit vorangehenden Projekt, dem Teamorientierten Praktikum, wurde zunächst das Hexagonal Image Processing Framework HIP aus der Programmiersprache Python in C überführt (genannt CHIP) und in die Komponenten Hexint, Hexarray und Hexsamp unterteilt. Die folgende Tabelle gibt einen Überblick über deren Funktionalität, wobei die Komponente Hexdisp zuletzt neu implementiert wurde, um eine Visualisierung des hexagonalen Rasters mittels OpenGL [Khr16] zu ermöglichen.

Komponente	Funktionalität
Hexint	Speicherung einzelner Bildpunkte und deren Adressen innerhalb der Spiralarchitektur des Frameworks. Implementiert die Funktionalität zum Addieren und Multiplizieren von Hexints sowie die zur Bestimmung von deren kartesischen Koordinaten.
Hexarray	Speicherung von 7^n Hexints in einem Array. Ermöglicht die Ein- und Ausgabe von Hexints und Sub-Hexarrays sowie deren Manipulation.

Hexsamp	Fasst die Funktionalität von Hexint und Hexarray zusammen. Implementiert die Transformation mittels bilinearer (*BL*) und bikubischer Interpolation (*BC*). Die Menge der zur Interpolation zu verrechnenden Bildpunkte ist hierbei über einen Radius anzugeben.
Hexdisp	Visualisierung eines Hexarrays. Weiterhin sind Transformationen wie Translationen, Rotationen und Skalierungen möglich. Eine Ausgabe der Hexarrays erfolgt mittels Screenshot-Funktion (Beispiel siehe Abbildung 1 (e)).

Tabelle 1: Komponenten CHIP-Framework

Die folgende Abbildung zeigt dazu das Ergebnis eines Testdurchlaufes für *BC* (b) mit dem Testbild Lena (a). Hier ist bereits der typische Rand des Hexarrays zu erkennen, wie er entsteht, wenn die Größe dieses, entsprechend der verwendeten Ordnung, nicht genügt, um sämtliche originalen Daten zu umfassen.

(a) Originalbild ($305 \times 297 = 90585$ Bildpunkte)

(b) Nach der Rücktransformation ($7^5 = 16807$ Hexints)

Abbildung 4: Testdurchlauf CHIP-Framework

Optimiert wurde CHIP durch die bereits in Kapitel 2 eingeführten Ansätze, wie Vorberechnungen, Verbesserung der Interpolationsmethoden und Addition von Hexints. Eine Parallelisierung ließ sich zudem an der Stelle der Rücktransformation mittels Pthreads [Bar15] realisieren.

3.2 Verfahren zur Einzelbild- und Videoverarbeitung

Die folgenden Punkte geben einen Überblick zu den in der Bachelorarbeit neu implementierten bzw. erweiterten Verfahren auf Basis des hexagonalen Rasters:

- Kantendetektion: Vergleicht die Farbwerte benachbarter Hexints und identifiziert über die entsprechenden Differenzen hochfrequente Bildinhalte wie beispielsweise Kanten. Niederfrequente sowie graduelle Farbverläufe werden hingegen abgeschwächt oder vollständig eliminiert. Die zum Einsatz kommenden Verfahren lassen sich weiterhin entlang der drei Achsen im Hexagonalen ausrichten und auch untereinander kombinieren, sodass Kanten verschiedener Richtungen jeweils optimaler bestimmt werden können [Sch16, S. 85].

- Skalierung: Zur Anpassung der Größe eines Bildes wurden eine Reihe ganzzahliger Verfahren implementiert. Diese bilden für die Hochskalierung immer 7^n Hexints auf 7^{n+1} ab und umgekehrt für die Herunterskalierung, wobei die Farbwerte der neuen Sub-Hexarrays mittels Durchschnittsbildung berechnet werden (nach [Sch16, S. 20 f.]).

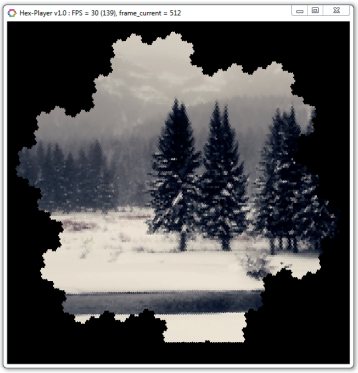
- Bewegungsschätzung und -ausgleich: Das Verfahren der Schätzung erkennt Bereiche mit Bewegung im Verlauf von zwei aufeinanderfolgenden Frames in Videomaterial und ordnet diesen einen Vektor zu. Die Bestimmung von Ähnlichkeiten erfolgt hierbei mittels der Vergleichsmetriken SAE, MAE, MSE, RMSE, PSNR, SSIM und DSSIM [Sch16, S. 79]. Durch den anschließenden Ausgleich erfolgt letztendlich eine Datenkomprimierung, sodass nur noch die alten Farbwerte und die Bewegungsvektoren zu speichern sind.

3.3 Integration in x264 – x264HMod

Zwecks Steuerung der integrierten Verfahren zur Einzelbild- und Videoverarbeitung wurde zunächst die Parameterübergabe von x264 selbst so modifiziert, dass sich diese auch über ihre jeweiligen Schalter ansteuern lassen. Die Ein- und Ausgabe, welche wie in Kapitel 1 beschrieben realisiert wurde, befindet sich hingegen noch in ihrem ursprünglichen Zustand.

Da x264 nur Ausgaben erzeugen kann, welche auf quadratischen Gruppen von Pixeln basieren (nach [Sch16, S. 39]), war es im Anschluss jedoch nicht möglich, direkt die hexagonalen Daten mit denen von x264 zu vereinbaren. Weiterhin müsste ein passendes Videoformat und Programm zur Wiedergabe geschaffen werden, um die entsprechenden Hexarrays adäquat darstellen zu können.

Der Grundstein hierfür wurde mit dem Tool „Hex-Player“ gelegt, welcher die durch x264-HMod ausgegebenen Hexarrays einlesen und auf Basis der Funktionalität von Hexdisp visualisieren kann. Abbildung 5 (a) zeigt dessen Oberfläche und rechts davon einen Ausschnitt dieser mit den erkennbaren Hexints des Frameworks (b). Diese ermöglicht neben der Wiedergabe des Videos auch die Steuerung der FPS-Zahl (frames per second – engl. für Bilder pro Sekunde) und einen manuellen Wechsel zwischen den einzelnen Frames.



(a) Oberfläche

(b) Ausschnitt aus (a)

Abbildung 5: Tool Hex-Player (Testvideo „Winter“ [Xip15])

4 Evaluation

Um die verschiedenen implementierten Interpolationsmethoden sowie die Verbesserungen der Transformation selbst, in und von dem hexagonalen Rasterformat, auf Qualität prüfen zu können, werden in den folgenden Abschnitten jene untereinander und auch mit einer aktuellen, auf quadratischen Rastern zum Einsatz kommenden, verglichen und adäquat beurteilt. Des Weiteren erfolgt eine Gegenüberstellung der ursprünglichen Version des Frameworks mit der Optimierten.

4.1 Hexagonale Bildtransformation

Die folgende Abbildung zeigt die Vergleichsergebnisse zu den beiden Interpolationsmethoden *BL* und *BC* über das Testset der Bilder nach [Web16]. Zum Einsatz kommen die Metriken MAE, MSE, PSNR und PHASH [CU10, KS10], wobei es sich, ausgenommen von PSNR, um Verzerrungsmaße handelt, welche mit größeren Werten auf Unterschiede hinweisen.

Erkennbar ist, dass sich beide Interpolationsmethoden trotz unterschiedlicher Berechnung nur geringfügig voneinander unterscheiden. Lediglich die Metrik PHASH, welche strukturelle Charakteristika mit berücksichtigt, weist einige inhaltsbedinge Unterschiede auf. Dies ist der Fall, da durch die lineare und kubische Funktionalität beider Verfahren, entsprechende Zwischenwerte jeweils optimaler dargestellt werden können. Hierzu zählen z. B. annähernd polynomielle Farbverläufe, welche sich nur schlecht oder gar nicht durch eine lineare Funktion approximieren lassen.

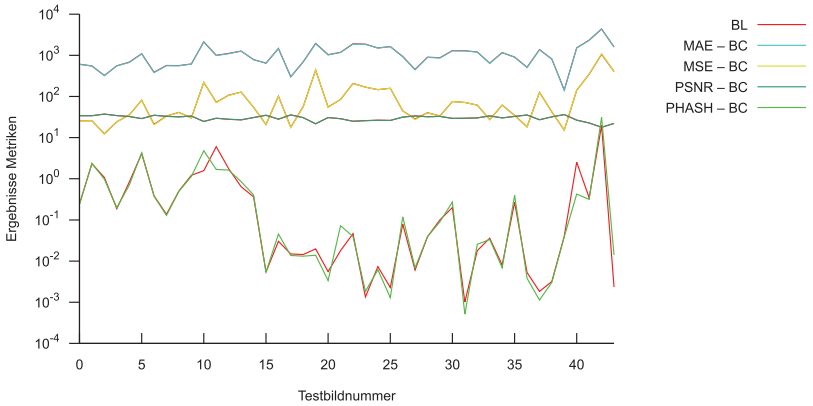


Abbildung 6: Vergleich Interpolationsmethoden mit Ausschnitt Testset

Zur weiteren Überprüfung der Transformationsqualität wurden zudem ihre Stadien miteinander verglichen. Abbildung 7 zeigt die Ausgabe der Rücktransformation im HIP-Framework (b), in CHIP mit zusätzlicher Interpolationsnormalisierung (c) und eine Weitere, für welche der Radius der Interpolation erweitert wurde, sodass eine Art Deblocking-Effekt eintritt (d). Bei (a) handelt es sich hingegen um das mittels des Softwarepaketes ImageMagick auf die Anzahl der Bildpunkte im Hexagonalen skalierte Originalbild. Diese Skalierung verwendet ein auf quadratischem Raster basierendes bilineares Verfahren zur Zwischenwertbestimmung (nach [Thy12]).

Hier ist zunächst erkennbar, dass der zeilenweise zyklische Fehler (b) durch die Normalisierung der Interpolationsmethoden (c) eliminiert werden konnte. Durch einen Vergleich der Ergebnisse beider Rasterformate ist des Weiteren feststellbar, dass die hexagonalen Verfahren (c und d) bedeutend weniger auffällige Artefakte und Verpixelungen erzeugen, was im Besonderen durch die homogenen Pixelabstände im Hexagonalen begünstigt wird.

4.2 Optimierung CHIP

Zur Auswertung der vorgenommenen Optimierungen kommen die drei Stadien der originalen Version des CHIP-Frameworks (genannt *Orig*), der mittels Vorberechnungen sowie

(a) Interpoliertes Original (b) Ausgabe HIP (c) CHIP mit Normalisierung der Interpolation (d) Erweiterter Radius mit Hexints 14 und 15

Abbildung 7: Vergleich bilineare Interpolation (Radius 1.0)

Verbesserung der Multiplikation von Hexints Optimierten (*Opt*), und eine Weitere, welche verschiedene durch den Compiler gesteuerte Optimierungsmethoden [Fre16] mit einbezieht (*Comp*), zum Einsatz.

Tabelle 2 zeigt die Ausführzeiten der Hin- und Rücktransformation für jeweils die Hexarray-Größen der Ordnung 5 bis 7, welche im Quadratischen die Auflösungen von ca. 305×297 bis 2141×2061 Pixel einschließen.[1]

Erkennbar ist, dass besonders die Rücktransformation mit zunehmender Ordnung von den einzelnen Optimierungsschritten profitiert. So entspricht beispielsweise der Schritt von *Orig* auf *Comp* für die Ordnungen 5 bis 7 ungefähr einer Beschleunigung um Faktor 57, 133 und 318. In HIP beträgt im Vergleich dazu die Ausführzeit beider Transformationen zusammengenommen bis zu 31 Stunden (Faktor 23000 langsamer).

Weiterhin konnten bei der Videoverarbeitung mittels x264HMod Geschwindigkeiten von bis zu 250 FPS gemessen werden, was dem einer relativ passablen Echtzeitfähigkeit entspricht.[2]

	Hintransformation			Rücktransformation		
Ordnung	5	6	7	5	6	7
Orig	0.219	1.513	10.86	4.955	85.24	1528
Opt	0.009	0.051	0.294	0.119	0.887	6.379
Comp	0.004	0.024	0.125	0.087	0.641	4.798

Tabelle 2: Optimierungsschritte CHIP

[1] CPU: AMD Phenom(tm) II X4 965 Processor, BogoMips: 7200, CPU Load: 99 % (ein Kern), VmSize: ca. 813 MiB (Max. für Ordnung 7)

[2] Hintransformation bei Ordnung 5 mit Ausgabe hexagonaler Daten

5 Zusammenfassung und Ausblick

Ziel war es, das Hexagonal Image Processing Framework CHIP, welches bereits die Funktionalität der Transformation, in und von dem hexagonalen Rasterformat dieses, bereitstellte, sowie dessen bildverarbeitenden Verfahren zu erweitern, optimieren und entsprechend zu evaluieren. Weiterhin sollte der Einbau in x264HMod, die hexagonale Erweiterung des Encoders x264, zur Videoverarbeitung erfolgen.

Neben einzelner hexagonaler Verfahren, wie Kantendetektion, Skalierung, Bewegungsschätzung und -ausgleich, wurde das Framework um die Komponente Hexdisp erweitert, welche eine Visualisierung des hexagonalen Rasters ermöglicht. Optimierungen und Verbesserungen ließen sich hingegen mittels verschiedener transformationsrelevanter Methoden und einer Parallelisierung realisieren.

Durch sämtliche vorgenommenen Optimierungen konnte während der Bewertung der einzelnen Erweiterungsstufen eine Beschleunigung der Ausführzeiten von bis zu $318n$-mal schneller als die originale Variante sowie ca. $23000n$-mal schneller im Vergleich zur ursprünglichen Python-Version des Frameworks gemessen werden ($n = \#\textit{Rechenkerne}$). In x264HMod sind zudem Verarbeitungsgeschwindigkeiten von bis zu 250 Einzelbildern pro Sekunde möglich.

Als problematisch erwies sich in diesem Zusammenhang die direkte Ausgabe der hexagonalen Daten in Form eines Videos. Auf Grund der Inkompatibilität beider Rasterformate resultierte letztendlich die Konsequenz, dass ein neues Videoformat und Wiedergabegerät geschaffen werden müsste, um diese adäquat visualisieren zu können. Der Grundstein hierfür wurde mit dem Tool „Hex-Player“ gelegt, welcher diese Funktionalität bereitstellt.

Zur Evaluierung der Transformation wurden die zum Einsatz kommenden Interpolationsmethoden untereinander und auch mit einer aktuellen, auf Viereck-Rasterformat basierenden, verglichen und entsprechend bewertet. Wie die Ergebnisse zeigen, bietet nicht nur das hexagonale Raster selbst, sondern auch die darauf aufbauenden Verfahren das Potential, qualitativ überzeugende Resultate zu erzielen. Für mögliche weiterführende Projekte lassen sich die Vorteile dieses untersuchen, wenn direkt aufgenommen durch einen hexagonalen Sensor oder ein angemessenes Alternativsystem.

Einen weiteren Punkt stellt die Implementierung eines hexagonalen Dateiformates für x264HMod und Hex-Player dar, wobei darüber hinaus auch noch diverse abhängige Methoden zur weiteren Datenkompression zu entwickeln sind. Hierfür wurde bereits ein erster Prototyp implementiert, welcher die Daten des Frameworks mit Hilfe der FFmpeg C API in einem unkodierten MKV-Containerformat abspeichert.

Acknowledgement

Diese Arbeit wurde teilweise in der InnoProfile-Transfer-Initiative *localizeIT* (Förderkennzeichen 03IPT608X) erstellt, das vom Bundesministerium für Bildung und Forschung (BMBF) gefördert wird.

Literatur

[Bar15] BARNEY, Blaise: *POSIX Threads Programming*, 14.08.2015. https://computing.llnl.gov/tutorials/pthreads/, abgerufen am 20.03.2016

[CU10] CONCI, A. ; ULYSSES, J.N.: *Measuring Similarity in Medical Registration*. In: *IWSSIP 2010 – 17th International Conference on Systems, Signals and Image Processing*, 17. – 19. Juni 2010

[Fre16] Free Software Foundation: *GCC online documentation – 3.10 Options That Control Optimization: -Ofast*, 2016. https://gcc.gnu.org/onlinedocs/gcc/Optimize-Options.html, abgerufen am 20.03.2016

[Gol69] GOLAY, Marcel J. E.: *Hexagonal Parallel Pattern Transformations*. In: *Computers, IEEE Transactions on (Volume: C-18, Issue: 8)*, August 1969, S. 733 – 740. http://dx.doi.org/10.1109/T-C.1969.222756

[Khr16] Khronos Group: *OpenGL Overview*, 2016. https://www.opengl.org/about/, abgerufen am 20.03.2016

[KS10] KLINGER, Evan ; STARKWEATHER, David: *pHash – Design & Validation*, 2010. http://www.phash.org/docs/design.html, abgerufen am 20.03.2016

[MS05] MIDDLETON, Lee ; SIVASWAMY, Jayanthi: *Hexagonal Image Processing: A Practical Approach*, 2005. – ISBN 978-1-84628-203-4

[Nel89] NEL, A. L.: *Hexagonal image processing*. In: *Communications and Signal Processing*, 23. Juni 1989, S. 109 – 113. http://dx.doi.org/10.1109/COMSIG.1989.129027

[Sch16] SCHLOSSER, Tobias: *Entwurf und Implementierung von Optimierungs- und Funktionserweiterungen der hexagonalen Bildrasterung in der Videokompressionssoftware x264HMod*, Technische Universität Chemnitz, Juniorprofessur Media Computing, Bachelorarbeit, Januar 2016

[Thy12] THYSSEN, Anthony: *Interpolative Resize – Resize using an Interpolation Method*, 10.10.2012. http://www.imagemagick.org/Usage/resize/#interpolative-resize, abgerufen am 20.03.2016

[Web16] WEBER, Allan: *The USC-SIPI Image Database – Volume 3: Miscellaneous*. http://sipi.usc.edu/DATABASE/database.php?volume=misc, abgerufen am 20.03.2016

[Xip15] Xiph.org Foundation: *Xiph.org Video Test Media [derf's collection]*, 23.12.2015. https://media.xiph.org/video/derf/, abgerufen am 20.03.2016

Entwicklung eines webbasierten Curve-fitting Tools für komplexe Multiparameter-Funktionen

Titus Keller, Danny Kowerko und Marc Ritter

Junior Professorship Media Computing
Technische Universität Chemnitz
09107 Chemnitz, Germany
titus.keller@s2012tu-chemnitz.de

Abstract: Dieser Beitrag stellt eine Webanwendung für Regressionsanalyse vor, die eine Reihe von Funktionalitäten enthält, an denen es den Standardprogrammen vieler Hersteller von Standalone-Programmen ebenso wie Webanwendungen mangelt. Dazu gehört beispielsweise das automatisierte Finden sinnvoller Startparameter sowie eine Weiterverarbeitung der Fit-Ergebnisse inklusive Fehlerfortpflanzung. In Kombination mit der intuitiven Import- und Exportfunktionalität wird dieses Webtool attraktiv für eine breite Anwenderschaft in den Natur-, Ingenieurs- und Lebenswissenschaften. Am Beispiel von thermischen DNS-Schmelzkurven wird demonstriert, wie selbst Gleichungen mit mehr als sechs Parametern durch nicht-lineare Regression zuverlässig an experimentelle Daten angepasst werden können. In das Web-Interface wurden Regressionsalgorithmen aus Java, Matlab und Octave implementiert, um deren Funktionalität hinsichtlich Rechengeschwindigkeit und Genauigkeit zu evaluieren.

1 Einleitung

Regressionsanalyse ist ein Teilgebiet der Mathematik, bei dem es darum geht Beziehungen von abhängigen und unabhängigen Variablen z.B. mittels mathematischer Gleichungen zu modellieren [Bac06]. Zwar wird der englische Begriff*Curve-fitting* zum Teil synonym zu Regressionsanalyse verwendet, aber es gibt auch Definitionen, in denen der Begriff die Teilgebiete Interpolation und Glättung beinhaltet [Gue13]. Im folgenden Text wird Curve-fitting mit Regressionsanalyse gleichgesetzt.

Ein allgemeiner Anwendungsschwerpunkt für dieses Verfahren ist die statistische Analyse. Besonders wenn es darum geht aus großen Datensätzen brauchbare Informationen zu gewinnen, ist Curve-fittng ein nützlicher Ansatz und lässt sich somit auch als eine Technik des Data-Minings verstehen [HMS01]. Dementsprechend findet das Verfahren zum Beispiel in den Geistes- und Sozialwissenschaften oder der Ökonomie Verwendung [KKNR13, Ram06].

$$y = f(x, a_n) + c\,, \tag{1}$$

wo die unabhängigen Parameter a_n durch Approximation einer Modellkurve an experimentell ermittelte Daten gewonnen werden. Bei der abhängigen Variable x kann es sich

um verschiedenste physikalische Größen handeln, zu denen u.a. Temperatur, Zeit, Strom, Spannung, Abstand oder Geschwindigkeit zählen.

Zur erfolgreichen computergestützten Ausführung einer Regressionsanalyse werden sinnvolle Startwerte für die Parameter a_n benötigt. Bei komplexen Funktionen kann es notwendig sein Startwerte vorab sinnvoll bzw. modellbasiert herzuleiten. Als Fallbeispiel sollen hier DNS (Desoxyribonukleinsäure)-Schmelzkurven dienen. Diese Methode gehört zu den Standardverfahren der Biochemie seit den 1970er Jahren, wenn es darum geht die chemische Stabilität eines DNS-Doppelstranges zu bestimmen. Dabei wird mittels UV-Absorptionsspektroskopie die Absorption der DNS als Funktionen der Temperatur gemessen und die thermodynamische Stabilität mit einer van't Hofft-Analyse bestimmt [FF77].

Gemäß dieser Theorie sind mathematische Gleichungen für DNS-Schmelzkurven mit Molekularitäten $1 \leq n \leq 4$ schon seit langem bekannt [FF77, BKS15], werden jedoch nur selten in der Regressionsanalyse verwendet. Ein möglicher Grund dafür ist, dass die entsprechenden expliziten Gleichungen mindestens sechs Parameter beinhalten. Die computergestützte Anpassung experimenteller Daten durch Funktionen mit sechs oder mehr unabhängigen Parametern ist anspruchsvoll und kann je nach Datenlage auch uneindeutig und damit unbrauchbar werden.

In diesem Artikel stellen wir ein Curve-fitting Tool vor, das die Regressionsanalyse erleichtern soll, indem Startwerte für die Parameter a_n autonom vor der eigentlichen Kurvenanpassung ermittelt werden. Als Modellsystem dient hier die thermische Schmelzkurvenanalyse von kurzen Doppelstrang-DNS-Molekülen.

1.1 Thermische Analyse von DNS-Schmelzkurven als Fallbeispiel.

Mathematisch werden thermische DNS Schmelzkurven wie folgt beschrieben:

$$A(T) = (m_1 \cdot T + n_1)\theta(T) + (m_2 \cdot T + n_2)(1 \quad \theta(T)) \tag{2}$$

Der Term $\theta(T)$ ist eine Funktion der Molekularität n, der freien Enthalpie ΔH^0 und der Schmelztemperatur T_m, detailliert beschrieben von Albrecht Böttcher und Kollegen [BKS15]. Im Fall von n = 2 ist $\theta(T)$ die Fraktion von als Doppelstrang vorliegender DNS.

Bei der DNS Schmelzkurvenanalyse wird die optische Absorption (A) einer in Puffer gelösten DNS-Probe (meist bei 260 nm Wellenlänge) als Funktion der Temperatur gemessen (siehe Abbildung 1). Im 2-Zustands-Modell liegt die DNS wie links in Abbildung 1 zu sehen als Doppelstrang (ds) oder Einzelstrangform (ss) (rechts im Bild) vor. Die Schmelztemperatur T_m zeigt an, bei welcher Temperatur ds und ss im Verhältnis 1:1 vorliegen. Die Abbildung wurde von einem DNA-Duplex-Stabilitäts-Tutorial der Firma ATDBio Ltd. entnommen. Für die erfolgreiche Anpassung von Gleichung (1) an experimentell bestimmte Messdaten müssen die Startwerte für m_1, n_1, m_2 und n_2 hinreichend genau bekannt sein. Daher ist es notwendig sie vorab durch lineare Regressionsanalyse mittels der Formeln

$$A(T) = m_1 \cdot T + n_1 \tag{3}$$

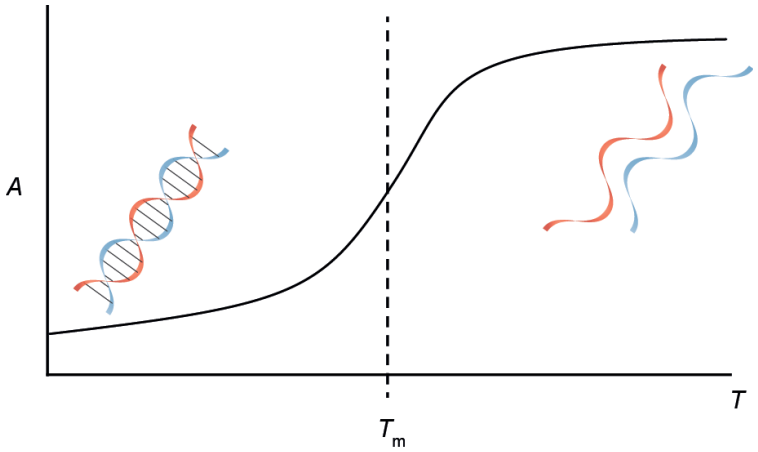


Abbildung 1: Schema einer thermischen Schmelzkurve von DNS: Es zeigt die Absorption A als Funktion der absoluten Temperatur T. (aus: Tutorium ATDBio. Ltd.[1])

bzw.

$$A(T) = m_2 \cdot T + n_2 \tag{4}$$

zu bestimmen. Dazu muss jedoch der Datenraum auf den jeweils linearen Bereich bei kleinen bzw. großen Temperaturen eingeschränkt werden. Es sind also mehrere Teilschritte notwendig, deren Automatisierung in vielen Regressionstools nicht ad-hoc bewerkstelligt werden kann. In dieser Arbeit wird eine Webapplikation vorgestellt, die diese Teilschritte vereinheitlicht und automatisiert.

Im Zusammenhang damit wird eine Betrachtung von Standalone- sowie Web-Softwarelösungen durchgeführt. Der Fokus liegt hierbei aber auf Letzteren und soll im Folgenden etwas genauer beleuchtet werden.

1.2 Stand der Technik existierender Curve-fitting (Web)-Applikationen

Momentan verfolgen viele Softwarelösungen im Bereich des Curve-fitting klassische Plug-in-Realisierungsansätze. Häufig wird zum Lösen der Problematik Mathematik-Software, wie z.B. MATLAB (The MathWorks, Inc.), Origin (OriginLab Corporation) oder GNU Octave verwendet. Aber es existieren auch Standalone-Implementierungen oder Realisierungen im Rahmen größerer Softwarepakete wie Excel oder ImageJ. Um die Programme tatsächlich nutzen zu können, sind einige Anforderungen zu berücksichtigen:

- Das Programm muss heruntergeladen/installiert werden.

[1] `http://www.atdbio.com/content/53/DNA-duplex-stability`, 23.03.2016

Eigenschaft	Erläuternde Frage
Proprietär	Ist die Applikation kommerziell?
Tutorial/Hilfe	Existiert eine Art von Hilfe, welche die Bedienung der Seite erläutert?
Eingabeoptionen	Wie präzise kann der Nutzer die Eingabeoptionen für den Curve-fit, wie z.B. Minimal- und Maximalwert der Parameter, festlegen?
Grafische Ausgabe	Wird das Ergebnis grafisch dargestellt?
Ergebnis-Export	Ist es möglich das Ergebnis zu exportieren, z.B. als PDF?
Fehlermaße	Werden Fehlermaße angezeigt?
Post-processing	Lassen sich die Fitergebnisse mathematisch weiterverarbeiten?

Tabelle 1: Vergleichskriterien für webbasierte Curve-fitting Applikationen.

- Das Programm ist nur auf bestimmten Plattformen und Betriebssystemen verfügbar.
- Gegebenenfalls sind Lizenzen zu erwerben.
- Tendenziell geht mit derartigen Programmen eine eingeschränkte Nutzerfreundlichkeit einher.
- Ein Teilen von (Zwischen-)Ergebnissen ist erschwert.

Diese Problematiken lassen sich prinzipiell durch die Verwendung webbasierter Implementierungsansätze lösen. Daher ergibt sich auch die Motivation dieser Arbeit. Infolgedessen wurde recherchiert und analysiert, ob und welche Softwarelösungen dieser Art bereits existieren. Dabei wurden insgesamt 24 entsprechende Applikationen ermittelt. Bei der Erstellung eines Überblicks in Hinsicht auf deren Inhalt und Funktionalität werden im Folgenden fünf repräsentative Webtools ausgewählt und genauer untersucht. Ein systematisches Vorgehen wird durch die in der Tabelle 1 notierten Vergleichskriterien ermöglicht.

Betrachtungsgegenstand hierbei sind die Webseiten *fitteia, WolframAlpha, arachnoid, mycurvefit* und *statpages*, deren Analyseergebnisse in Tabelle 2 dargestellt werden. Die Markierungen in dieser entsprechen einer groben Untergliederung; das heißt "-" steht für nicht beziehungsweise wenig erfüllt, wohingegen ein Häkchen bzw. "tw." meint, dass die Anforderungen vollständig bzw. nur teilweise abgedeckt werden.

Aus der Übersicht lässt sich resümieren, dass zwar eine Vielzahl von Webtools existiert, diese jedoch mehrere Unzulänglichkeiten in verschiedenen Teilbereichen aufweisen. Zusätzlich ist zu erwähnen, dass schwer fassbare Kriterien wie Bedienbarkeit (hier ließe sich zum Teil auf Normen der Software-Ergonomie zurückgreifen z.B. EN ISO 9241

[2] `http://fitter.ist.utl.pt/`, 24.03.2016
[3] `https://www.wolframalpha.com/`, 24.03.2016
[4] `http://arachnoid.com/polysolve/`, 24.03.2016
[5] `http://mycurvefit.com/`, 24.03.2016
[6] `http://statpages.info/nonlin.html`, 24.03.2016

Website	fitteia[2]	WolframAlpha[3]	arachnoid[4]	mycurvefit[5]	statpages[6]
Proprietär	✓	-	✓	-	✓
Tutorial/Hilfe	✓	✓	✓	-	✓
Eingabeoptionen	tw.	✓	-	tw.	tw.
Grafische Ausgabe	✓	✓	✓	✓	-
Ergebnis-Export	✓	tw.	tw.	✓	-
Fehlermaße	-	tw.	✓	✓	tw.
Post-processing	-	✓	-	-	-

Tabelle 2: Eigenschaften ausgewählter Curve-fitting Webapplikationen.

[dN95]) oder optischer Eindruck bei diesen Betrachtungen nicht explizit einbezogen wurden. So ist z.B. für die Nutzung der allgemeinen Mathematiksoftware WolframAlpha eine Kenntnis der entsprechenden Syntax notwendig, wobei Fehlermaße nur durch weitere nutzergesteuerte Eingaben berechnet werden können.

Aus (natur-)wissenschaftlicher Sicht erscheinen besonders die Eigenschaften eines vollständigen Ergebnis-Exports mit allen Fehlermaßen und die Möglichkeit einer Datenweiterverarbeitung (Post-processing) relevant. Gemessen an allen Anforderungen, schneidet WolframAlpha mit einigen Einschränkungen am besten ab, hat jedoch den Nachteil nicht kostenfrei erhältlich zu sein. Daraus ergibt sich die Motivation eine eigene umfassende Anwendung zu entwickeln, welche die ausgewählten und weitere Kriterien in einer zufriedenstellenden Art und Weise erfüllt.

2 Ergebnisse und Diskussion

Im Folgenden wird die eigens entwickelte Curve-fitting Webanwendung vorgestellt und deren technische Realisierung zusammengefasst. Die aktuelle Entwicklungsversion ist verfügbar unter[7].

Eine Bildschirmkopie (Screenshot) der Anwendung ist in Abbildung 2 zu sehen. Es handelt sich um eine Einzelseitenanwendung mit 4 Teilbereichen:

- (links, oben) Datenimport; Definition der Fitfunktion; Button für vordefinierte Konstanten, Funktionen, Datensätze
- (rechts, oben) Grafische Anzeige des Datensatzes und der an diesen angepassten Funktion in einem interaktiven Graph
- (links, unten) Fitparameter und Fehler, sowie Gütemaße des Fits
- (rechts, unten) Hier werden die Post-processing-Funktionen angezeigt. Alternativ wird hier das Residuum angezeigt.

[7] `http://skinner.informatik.tu-chemnitz.de:61580/`, 24.03.2016

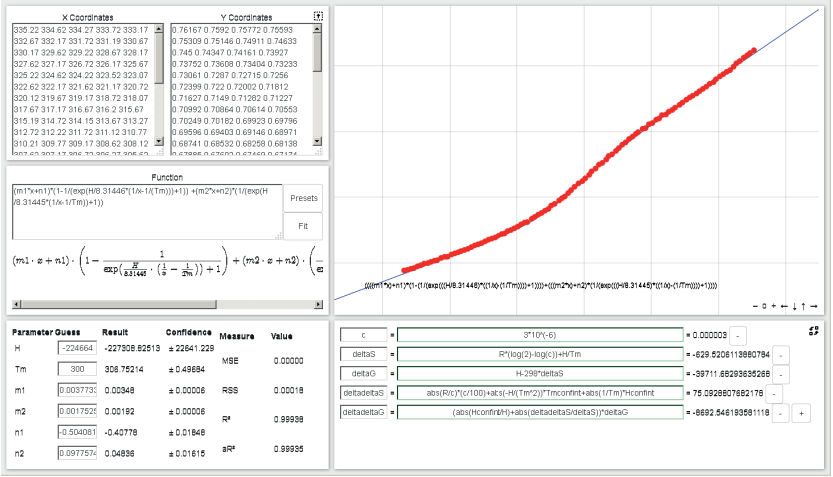


Abbildung 2: Grafische Nutzeroberfläche der entwickelten Webapplikation zur Regressionsanalyse. Verwendete Daten entstammen der Publikation von Kowerko und Kollegen [KKS+15]. Die Webseite ist abrufbar, siehe Fußnote [8].

Die primären Funktionalitäten der Website sind in Tabelle 3 zusammengefasst. Es wurde insbesondere darauf geachtet, dass Fehlermaße und eine Datenweiterverarbeitung (Post-processing) zur Verfügung stehen, die es ermöglicht Fehlerrechnung inklusive Fehlerfortpflanzung mit den bestimmten Genauigkeiten der Fitparametern durchzuführen.

Die relevanten biochemischen Informationen sind die freie Enthalpie ΔH^0 und die Schmelztemperatur T_m, welche in Abbildung 2 durch H und Tm wiedergebeben werden. Die konkreten Fitergebnisse unter Nutzung von Michael Thomas Flanagan's Java Scientific Library[9] ergeben

$$\Delta H^0 = \quad (227.3 \pm 22.6) kJ/mol$$

und

$$T_m = (306.8 \pm 0.5) K.$$

Unter Verwendung der bekannten Konzentration c = $3 \cdot 10^{-6}$ mol/l lassen sich die Werte und Fehler der freien Standardreaktionsentropiedifferenz und der freien Energiedifferenz entsprechend der Formeln im Post-processing-Fenster von Abbildung 2 (rechts, unten) ermitteln:

$$\Delta S^0 = \quad (629.5 \pm 75.1) J/molK$$

und

$$\Delta G^0 = \quad (39.6 \pm 8.7) kJ/mol.$$

[8] http://skinner.informatik.tu-chemnitz.de:61580/#/56f4523493cca9f7b0ef142a, 24.03.2016

[9] http://www.ee.ucl.ac.uk/~mflanaga/java/, 19.03.2016

Website	Bewertung	Beschreibung
Proprietär	✓	Eine Veröffentlichung der Software in einem kommerziellen Rahmen ist momentan nicht vorgesehen.
Tutorial/Hilfe	-	Es werden keine weiterführende Erläuterungen oder Erklärungen zur Verfügung gestellt.
Eingabeoptionen	tw.	Individuelle Schätzungen für die Parameter sind übermittelbar, jedoch ist es z.B. nicht möglich Minimal- oder Maximalwerte einzugeben.
Grafische Ausgabe	✓	Das Ergebnis wird in einem interaktivem Graph angezeigt.
Ergebnis-Export	✓	Ein Export ist als Link oder als JSON/PDF-Dokument möglich.
Fehlermaße	✓	Die wichtigsten Fehlermaße für die Thematik, sowie das Residuum werden angezeigt.
Post-processing	✓	Auf Basis der Ergebnisse für die Parameter können automatisch weitere mathematische Berechnungen durchgeführt werden.

Tabelle 3: Eigenschaften der entwickelten Curve-fitting Webapplikation.

Die Formeln für ΔS^0 und ΔG^0 sind der Fachliteratur entnommen [FF77]. Die dazugehörigen Fehlerformeln wurden bestimmt nach dem Standardverfahren der physikalischen Fehlerrechnung inkl. Fehlerfortpflanzung [SKB+14].

Weiterhin lieferte die eingebundenen Funktion von Octave ebenfalls plausible Fitergebnisse. Mit der integrierten Matlab Methode und der ImageJ Curve-fit Bibliothek konnten bisher keine erfolgreichen Kurvenanpassungen durchgeführt werden.

2.1 Technologie

Bei der technologischen Umsetzung der Applikation wurde festgelegt, dass zuerst die Funktionalität umgesetzt werden soll, bevor die Entwicklung eines passenden Designkonzepts fokusiert und die Nutzerfreundlichkeit weiter optimiert wird. Erhofft wird dadurch beim späteren Entwerfen der optischen Repräsentation auf Erfahrungen, die bei der Umsetzung der Funktionalität zustanden kamen, zurückgreifen zu können.

Da Algorithmen zur Lösung der nicht-linearen Regression nur sehr geringfügig in JavaScript verfügbar sind, war es notwendig dieses Problem serverseitig zu lösen. Die Umsetzung dessen erfolgte unter Nutzung der Programmiersprache Java. Auf Grund deren weitverbreiteten Nutzung bieten viele Mathematik-Applikationen (z.B. Matlab, Scilab und Oc-

tave) entsprechende Schnittstellen an. Im konkreten Anwendungsfall dient Jetty als HTTP-Server und Servlet-Container. Um vom Client veranlasste Berechnungen auf dem Server durchzuführen wurde Jersey als Servlet eingebunden. Mit Hilfe dieser Bibliothek können auf REST basierende Webservices umgesetzt werden [Abt15]. Durch die Verwendung einer solchen zustandslosen Technologie lässt sich die Anwendung potentiell recht einfach durch Verwendung eines Lastverteilers horizontal skalieren.

So wurde eine Schnittstelle zur Durchführung von Curve-fit Berechnungen geschaffen. Als konkrete Implementationen hierfür wurden Softwarefunktionen von Matlab (Funktion "lsqcurvefit"), Octave (Funktion "leasqr" im Paket optim), ImageJ und Michael Flanagan's Java Scientific Library angebunden. Um zusätzlich ein Verbreiten von Fitergebnissen via Hyperlink zu ermöglichen, wurde eine MongoDB-Datenbank aufgesetzt. Diese nutzt als internes Format JSON, welches entsprechend mit dem Front-End harmoniert. Durch den dokumentorientierten Ansatz der Datenbank lässt diese sich durch sogenanntes sharding horizontal skalieren.

Auf der Seite des Clients wurden die Standard Webtechnologien HTML/CSS/JS verwendet. Zur Erleichterung der Arbeit und zur Minimierung des Implementierungsaufwands befindet sich AngularJS als allgemeines JavaScript MVC-Framework im Einsatz. Diese Bibliothek ist neben Ember.js und Backbone.js vermutlich das meistverwendete Framwork für diese Problematik.[10]

Der Model-View-Controller Ansatz soll dabei helfen den Code zu strukturieren. In diesen Zusammenhang können auch wiederverwendbare Komponenten erstellt werden, was z.B. die Wartung der Anwendung erleichtert. Unabhängig davon wäre ein Beispiel für die Anwendung von AngularJS das sogenannte data binding, welches es ermöglicht eine Verbindung zwischen JavaScript und HTML zu schaffen und so die Anzahl der manuell durchzuführenden DOM-Zugriffe minimiert [HG12]. Weitere wichtige Bibliotheken, die verwendet wurden, sind:

- JS Expression Evaluator zum Parsen, Bearbeiten und Auswerten mathematischer Funktionen.
- MathJAX zum Rendern der in ASCII eingegebenen Formel.
- JSXGraph zur grafischen Darstellung von Ergebnissen.
- JSPDF, um einen Export in Form einer übersicht im PDF Format umzusetzen.

2.2 Standardisierung und Nachhaltigkeit

Durch das Anbieten programmiersprachenübergreifender Regressionsalgorithmen ist es möglich, die jeweiligen Implementationen hinsichtlich Genauigkeit und Rechengeschwindigkeit gegeneinander zu evaluieren. Damit soll eine methodenspezifische Standardisierung vorangetrieben werden. Modellhaft wurden hier vier Curve-fitting Funktionen imple-

[10] `https://www.airpair.com/js/javascript-framework-comparison`, 24.03.2016

mentiert. Es stellte sich heraus, dass für den Beispieldatensatz mit ImageJ und Matlab keine, mit den Java-Funktionen von Flanagan und aus Octave jedoch zuverlässige Ergebnisse erzielt werden konnten. Die nachhaltige Nutzung bestehender Resourcen der OpenSource-Community wurde damit eindrücklich gezeigt.

Die Webapplikation ist damit nicht nur für Anwender, sondern auch für Entwickler interessant, da sich damit auch Modelle testen lassen, ganz konkret bspw. Curve-fit Implementationen aus anderen Toolboxen oder Programmiersprachen wie C++, Python (SciPy) oder R. Ein Vorteil der Webanwendung ist, dass sich Entwickler zunächst wenig oder gar nicht um Control und View kümmern müssen, sondern die allgemein gültigen Eingaben der grafischen Nutzeroberfläche nutzen können um Ihr Model zu testen. Derartige Erweiterungen an Methoden erlauben es Standards zu definieren, die sich auf hohe Funktionalität und Rechengeschwindigkeit beziehen.

3 Zusammenfassung

Die im Projekt entwickelte Curve-fitting-Webapplikation erlaubt es Startparameter durch Regression mit weiteren mathematischen Funktionen über einen Teilbereich der Daten zu finden und die Fitergebnisse mitsamt ihrer Fehler direkt weiter zu verarbeiten, um Zielgrößen zu bestimmen. Am Beispiel thermischer DNS-Schmelzkurvenanalyse wurde die Funktionalität des Tools mittels Matlab, Octave und Java-basierten Algorithmen demonstriert, was sie auch für Entwickler und den Bereich Methodenevaluation interessant macht. Es können selbst komplexe Multiparametergleichungen exakt durch explizite Gleichungen approximiert werden. Durch die Möglichkeit Nutzerfunktionen anzulegen ist die Anwendung nicht auf die hier aufgeführte Schmelzkurvenanalyse beschränkt, sondern hat das Potential eine breite Zielgruppe von Anwendern auf interdisziplinärem Gebiet in den Geistes-, Natur-, Ingenieurs- und Lebenswissenschaften zu finden. Die Kombination von Fehlerrechnung inkl. Fehlerfortpflanzung mit der Berichtsexportfunktion als PDF gestattet auch eine Anwendung zur Dokumentation im Laboralltag.

4 Ausblick

Ferner ist auf funktioneller Seite die Integration eines Evaluationsmoduls angedacht, das es ermöglicht Daten basierend auf benutzerdefinierten Funktionen zu simulieren und mit statistischem Rauschen zu überlagern. Damit wird es möglich sein, reale Daten zu modellieren und die Genauigkeit des Fits im konkreten Anwendungsfall vorherzusagen. Außerdem wird Mehrfachdatenimport und Export ermöglicht werden. Eine Weiterentwicklung zum universitätsweiten Webservice ist in Prüfung. Dieser könnte analog zu *dict.tu-chemnitz.de* oder *mail.tu-chemnitz.de* unter *curvefit.tu-chemnitz.de* erreichbar sein.

Die Bedienbarkeit soll vereinfacht und durch Hinzufügen eines Hilfsmodus verbessert werden, der dem Nutzer erklärt, welchen Hintergrund und Bedeutung die Elemente der Webseite haben. Darüber hinaus ist die Integration einer Feedbackfunktionen angedacht,

um formative Evaluationsergebnisse zu erhalten, was bedeutet, dass die Plattform während der Entwicklung kontinuierlich evaluiert und verbessert werden kann [Scr91]. Diese Eingaben können in der bereits angebundenen MongoDB Datenbank gespeichert werden. Ein benutzerfreundliches Design der Webseite wird in weiteren Studien durch Mockups evaluiert werden [ZC03]. An Hand dieser lassen sich zum Teil auch durch Programme, welche auf kognitiven Modellen basieren, automatisierte Vorhersagen über die Nutzer-Performance treffen. Eine solche Anwendung wäre z.B. CogTool [SCJB13].

Danksagung

Diese Arbeit wurde teilweise in der Initiative *localizeIT* (Fördernummer: 03IPT608X) durchgeführt, die durch das *Bundesministerium für Bildung und Forschung* (BMBF, Deutschland) im Programm *Unternehmen Region InnoProfile-Transfer* gefördert wird.

Literatur

[Abt15] Dietmar Abts. REST-basierte Web Services mit JAX-RS. In *Masterkurs Client/Server-Programmierung mit Java*, Seiten 277–330. Springer, 2015.

[Bac06] Klaus Backhaus. *Multivariate Analysemethoden eine anwendungsorientierte Einfhrung*. Springer, Berlin, 2006.

[BKS15] Albrecht Bttcher, Danny Kowerko und Roland KO Sigel. Explicit analytic equations for multimolecular thermal melting curves. *Biophysical chemistry*, 202:32–39, 2015.

[dN95] CEN-Comit Europen de Normalisation. *Ergonomische Anforderungen fr Brottigkeiten mit Bildschirmgerten Teil 10: Grundstze der Dialoggestaltung*. Februar, 1995.

[FF77] Marshall Fixman und Juan J. Freire. Theory of DNA melting curves. *Biopolymers*, 16(12):2693–2704, 1977.

[Gue13] Philip George Guest. *Numerical methods of curve fitting*. Cambridge University Press, Cambridge, 2013.

[HG12] Matthias Heinrich und Martin Gaedke. Data binding for standard-based web applications. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, Seiten 652–657. ACM, 2012.

[HMS01] David J. Hand, Heikki Mannila und Padhraic Smyth. *Principles of data mining*. MIT press, 2001.

[KKNR13] David G. Kleinbaum, Lawrence L. Kupper, Azhar Nizam und Eli S. Rosenberg. *Applied regression analysis and other multivariable methods*. Cengage Learning, Boston, MA, fifth edition. Auflage, 2013.

[KKS+15] Danny Kowerko, Sebastian L. B. Knig, Miriam Skilandat, Daniela Kruschel, Mlodie C. A. S. Hadzic, Lucia Cardo und Roland K. O. Sigel. Cation-induced kinetic heterogeneity of the intronexon recognition in single group II introns. *Proceedings of the National Academy of Sciences*, 112(11):3403–3408, Marz 2015.

[Ram06] Rodney Ramcharan. Regressions: Why Are Economists Obessessed with Them? *Finance Dev*, 43, 2006.

[SCJB13] Amanda Swearngin, Myra B. Cohen, Bonnie E. John und Rachel K. E. Bellamy. Human Performance Regression Testing. In *Proceedings of the 2013 International Conference on Software Engineering*, ICSE '13, Seiten 152–161, Piscataway, NJ, USA, 2013. IEEE Press.

[Scr91] Michael Scriven. *Evaluation thesaurus*. Sage, 1991.

[SKB+14] Wolfgang Schenk, Friedrich Kremer, Gunter Beddies, Thomas Franke, Petrik Galvosas und Peter Rieger. *Physikalisches Praktikum*. Springer Fachmedien Wiesbaden, Wiesbaden, 2014.

[ZC03] Jia Zhang und Jen-Yao Chung. Mockup-driven fast-prototyping methodology for web application development. *Software Practice & Experience Journal, 33 (13), 2003, pp.*, Seite 1251, 2003.

Entwicklung einer virtuellen Versuchsumgebung zur experimentellen Untersuchung von Raumorientierung und visueller Aufmerksamkeit

Sascha Jüngel, Frederik Beuth, Fred H. Hamker

Technische Universität Chemnitz, Künstliche Intelligenz
Strasse der Nationen 62, D-09111 Chemnitz, Deutschland
sascha.juengel@s2010.tu-chemnitz.de

Zusammenfassung. Was für Menschen kein Problem ist, stellt für Roboter eine Herausforderung dar: Den visuellen Raum in kürzester Zeit zu erfassen und relevante Informationen daraus abzuleiten. Ein visuelles System erhält so viele sensorische Daten, das eine clevere Fokussierung auf das Wesentliche nötig ist. Diese Fragestellungen werden im Europäischen Projekt Spatial Cognition erforscht, um diesen noch nicht vollständig verstandenen Prozess der Aufmerksamkeit und Raumwahrnehmung in Zukunft besser verstehen zu können. Die hier vorliegende Arbeit beschreibt die Erstellung einer virtuellen Szene mit vier verschiedenen Einzelszenarien für dieses EU Projekt, um verschiedene Aspekte der Raumwahrnehmung erforschen zu können. Der Schwerpunkt liegt dabei auf der Objektlokalisation, dem Aktualisieren räumlicher Kontextinformationen, dem aufmerksamkeitsbasierten Beobachten mehrerer Objekte und dem Erinnern an eine Position im Raum. Ein virtueller Agent (Kind) führt diese Aufgaben aus, wobei er über die Fähigkeiten verfügt Objekte anzuschauen und zu greifen. Weiterhin wird ihm, unter anderem durch eine implementierte Pfadsuche, die Möglichkeit gegeben sich in seiner virtuellen Umgebung (Kinderzimmer) zu bewegen. Bis in dem Forschungsprojekt Spatial Cognition entwickelte Gehirnmodelle den Agenten steuern, sind die Handlungsabläufe des Agenten vorgegeben, um die unterschiedlichen Szenarien durchführen zu können. Insgesamt liegt nun eine stabile, virtuelle Versuchsumgebung mit dynamischen Schnittstellen vor, die für alle nachfolgenden Forschungen in diesem Bereich verwendet werden kann.

Stichworte: Visuelle Aufmerksamkeit · Raumorientierung · Virtuelle Realität

1 Einleitung

Für Menschen ist es einfach ein Zielobjekt in einer unbekannten und unstrukturierten Umgebung zu greifen [Ant14]. Es erfordert allerdings die Implementierung komplexer Fähigkeiten, wie die Identifikation des Ziels im Sichtbereich und das präzise Steuern von Motoren um einen Roboter zu entwerfen, der die selbe Aufgabe ausführen kann. Technologien wie Videoüberwachung und visuell gesteuerte Roboter benötigen flexible visuelle Systeme um die Szene entsprechend der aktuellen Aufgabe zu interpretieren [Ham05]. Es ist naheliegend, dass Aufmerksamkeit eine wichtige Rolle beim Verstehen einer Szene spielt, da sie relevante Informationen priorisiert. Die zugrunde liegenden Prozesse dieser wurden jedoch noch nicht vollständig entdeckt. Was wir allerdings wissen ist, das wir durch Aufmerksamkeit in der Lage sind, Objekte oder Regionen im visuellen Raum zu überwachen und Informationen aus diesen zu ziehen [CA05]. Klassische Theorien über Aufmerksamkeit nahmen an, dass es ein einzelner ausgewählter Bereich fokussiert wird. Aber viele alltägliche Tätigkeiten wie beispielsweise das Aufpassen auf Kinder im Schwimmbad benötigen Aufmerksamkeit an mehreren Stellen. Und in der Tat haben auch Versuche im Labor gezeigt, dass wir die Fähigkeit haben, vier oder mehr Zielobjekte gleichzeitig zu verfolgen.

Im Rahmen des EU-Projektes Spatial Cognition soll die Orientierung im Raum und das Extrahieren relevanter Informationen aus der Umgebung erforscht werden [spa]. Ziel ist es, herauszufinden, wie das Aktualisieren räumlicher Kontextinformationen funktioniert und durch welche Mechanismen der Überblick über wichtige Umgebungsdetails beibehalten und Informationen aus diesen gewonnen werden. Hierbei hilft vor allem der Versuch diese Prozesse zu simulieren und innerhalb eines Gehirnmodells so weit nachzustellen wie sie bisher erschlossen sind. Ein solches virtuelles Gehirnmodell benötigt dafür jedoch eine virtuelle Versuchsumgebung, in der verschiedene, die visuelle Wahrnehmung untersuchende Experimente durchgeführt werden können. Eine solche Umgebung wurde im Rahmen dieser Arbeit erstellt. Als Basis diente die Unity-Engine, eine flexible Entwicklungsplattform zur Erstellung von 2D und 3D Spielen und anderen interaktiven Anwendungen. Später soll der an der TU Chemnitz entwickelte Neuro-Simulator ANNarchy (Artificial Neural Networks architect) verwendet werde um das Gehirn des Agenten zu simulieren.

Die Grundlagen des bereits vor dieser Arbeit existenten Frameworks zur Kopplung von VR und Agent werden im folgenden Kapitel beschrieben. Kapitel 3 beschäftigt sich mit der Erweiterung dieses Frameworks um diverse neue Funktionen, die Sensorik und Aktorik betreffen. Im nachfolgenden, 4. Kapitel werden verschiedene Problemstellungen der räumlich-visuellen Wahrnehmung erläutert und für jede von diesen wird ein Szenario innerhalb der virtuellen Versuchsumgebung bereitgestellt, um weitere Erkenntnisse auf diesem Gebiet gewinnen zu können. Da aktuell die Kontrolle über den Agenten, der diese Szenarien ausführt, noch nicht von einem Gehirnmodell übernommen wird, werden viele Abläufe vereinfacht, automatisiert oder nachgebildet um die Funktionalität der erstellten Versuchsumgebung zu demonstrieren. Diese Arbeit hat nicht direkt zum Ziel einen tieferen Einblick in die Prozesse der visuellen Wahrnehmung zu erhalten, sondern stellt viel mehr ein grundlegendes Framework zur Verfügung, um dieses Verständnis in zukünftigen Forschungsvorhaben erhalten zu können.

(a) **(b)**

Abbildung 1: (a) Die Virtuelle Umgebung, die für die Szenarien genutzt wird. (b) Der virtuelle Agent Felice.

2 Simulation eines neuro-kognitiven Agenten in der VR

Die Grundlage der in dieser Arbeit erstellten Szenarien bildet ein bereits in der Vergangenheit an der TU Chemnitz entwickeltes Framework. Der Zugang zum Spatial Cogniton Projekt wird in Zukunft auf der zugehörigen Homepage bereitgestellt (`https://www.tu-chemnitz.de/informatik/KI/projects/agents-vr`). Da der Lernprozess visueller Stimuli hauptsächlich während der ersten Lebensjahre auftritt, wurde Felice einem Kind nachempfunden. Dieses kann sich, unter Berücksichtigung von Kollisionen, in der Szene durch Laufen und Drehen bewegen, sowie Objekte greifen und anschauen. Die virtuelle Umgebung wurde passend zum Agenten gewählt und stellt ein Kinderzimmer da, welches neben anderen Einrichtungsgegenständen zwei Tische beinhaltet, auf denen sich verschiedene Objekte befinden, die in den einzelnen Szenarien von Bedeutung sind (Abb. 1).

Die VR läuft unter Windows und stellt alle sensorischen Daten, wie stereoskopische Bilder und Kollisionen bereit. Kern der in der VR verwendeten Skripte bilden ein Agenten- (*AgentScript*) und ein Umgebungsskript (*BehaviourScript*). Das Agentenskript steuert die Bewegungen des Agenten und verwaltet die Eingangsdaten wie zum Beispiel Kamerabilder. Allerdings wird es nicht direkt an das Agenten-Objekt angehängt, sondern ein von ihm abgeleitetes Skript, das die im ursprünglichen Agentenskript enthaltenen virtuellen Methoden überschreiben kann und so die Möglichkeit hat, das Verhalten jedes Agenten individuell anzupassen. Das Umgebungsskript ist für die Kontrolle der virtuellen Umgebung

zuständig. Sein abgeleitetes Skript wird an ein Master-Objekt angehängt, enthält Referenzen auf die Agenten sowie deren Agentenskripts und initialisiert diese. Neben diesen beiden Haupt-Skripten wurden noch diverse andere Skripte (beispielsweise zum Blenden verschiedener Greifanimationen und der Berechnung von Kollisionen), sowie Unity-Prefabs und die vom Agenten genutzten Animationen durch frühere Projekte bereitgestellt.

Die Agentenseite läuft unter Linux und ist eine C++ Implementierung. Aktuell werden dem Agenten von hier aus lediglich Nachrichten über das Netzwerk zugesendet um in Test-Szenarios die Funktionalität der VR zu testen. Später soll der an der TU Chemnitz entwickelte Neuro-Simulator ANNarchy (Artificial Neural Networks architect) verwendet werden um das Gehirn des Agenten zu simulieren [ann]. Dessen Kern-Bibliothek wurde in C++ geschrieben und kann entweder durch openMP oder CUDA parallelisiert werden. Er stellt außerdem ein Python-Interface für Netzwerke bereit.

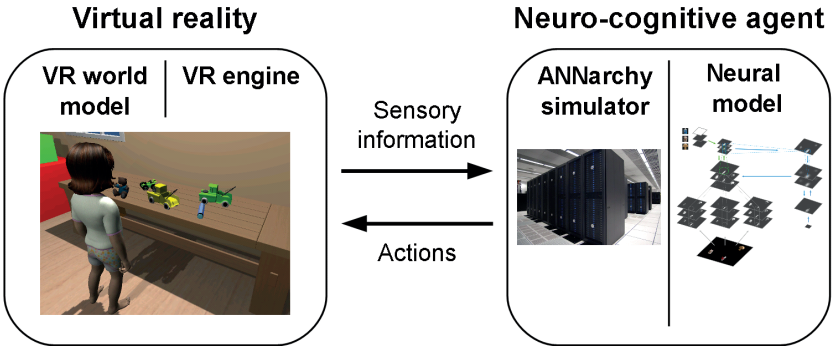


Abbildung 2: Das Framework um neuro-kognitive Agenten zu simulieren, besteht aus einer virtuellen Realität, in der sich der Agent befindet und dessen Gehirn.

Um die Kommunikation zwischen der VR- und Agentenseite (Abb. 2) zu ermöglichen, werden eine Netzwerkverbindung (TCP/IP) sowie das von Google Inc. entwickelte Datenformat Protocol Buffers (kurz protobuf) verwendet [pro]. Dabei handelt es sich um einen erweiterbaren, sprachen- und plattformübergreifenden Mechanismus zur Serialisierung, ähnlich XML. Genutzt wurde die C++ Implementierung. Protobuf ermöglicht das Erstellen verschiedener Nachrichten, die entweder von VR- zu Agentenseite gesendet werden können oder umgekehrt. Um die Test-Szenarien dieser Arbeit durchführen zu können, wurden einige der bereits in der Vergangenheit implementierten Protobuf-Nachrichten verwendet: *MsgAgentEyeMovement* lässt Felice die Augen bewegen. Als Input werden jeweils die Drehung um die y-Achse (pan) für das linke und rechte Auge sowie die Drehung um die x-Achse (tilt) für beide Augen entgegen genommen. Diese Nachricht wird vor allem dazu verwendet, den Blick des Agenten wieder in die Ausgangsposition (gerade nach vorne) zu richten. Um etwas anzuschauen und zu fixieren, wird die Nachricht *MsgAgentEyeFixation* verwendet, die das Fixations-Ziel in Weltkoordinaten entgegen nimmt. Der Blick des Agenten bleibt dann auf diese Position fixiert (auch wenn dieser sich be-

wegt), bis *MsgAgentEyeMovement* gesendet wurde. An eine Position greifen kann Felice durch die Nachricht *MsgAgentGraspPos*, die Kamerakoordinaten entgegen nimmt und diese dann ins Weltkoordinatensystem abbildet. Ein gegriffenes Objekt kann durch *MsgAgentGraspRelease* wieder losgelassen werden. Um die globale Ausgangssituation eines Szenarios herzustellen wird die Nachricht *MsgEnvironmentReset* verwendet. *MsgTrialReset* ist zum Zurücksetzen von Teilaufgaben gedacht und wird in den Szenarien dieser Arbeit zur Manipulation der Umgebung (beispielsweise dem Ein-/Ausblenden von Objekten) genutzt.

3 Erweiterte Sensorik und Aktorik

Im Rahmen dieser Arbeit wurden Sensorik und Aktorik des Agenten erweitert um zusätzliche Informationen zu erhalten bzw. neue Aufgaben durchführen zu können. Weiterhin wurde kleinere Fehler korrigiert.

Eine wichtige Neuerung sind die Augenbewegungen zwischen zwei Fixationsphasen (Sakkaden). Diese wurden entsprechend der Beschreibungen in [BH08] und [VWVO88] implementiert.

$$A = \frac{1}{1 - e^{-\frac{R}{m_0}}} \tag{1}$$

A: Skalierungsfaktor
R: Verschiebung auf der Retina
m_0: konst. ($m_0 = 7$)

$$V_{pk} = V_{max} * (1 - e^{-\frac{\phi}{n}}) + 50; \tag{2}$$

V_{pk} : höchste Augengeschwindigkeit
V_{max}: Maximale Geschwindigkeit ($V_{max} = 750$)
ϕ: Winkel der Augenbewegung ($0 \geq \phi \geq 60$) [deg]
n: konst. ($n = 15$)

$$Ecc = m_0 * \ln \frac{A * e^{\frac{V_{pk} * t}{m_0}}}{1 + A * e^{\frac{V_{pk} * t - R}{m_0}}} \tag{3}$$

Ecc : Momentane Augen-Exzentrizität (= S(t) in [VWVO88])

Mit *MsgEyePosition* und *MsgHeadMotion* wurden zwei Nachrichten implementiert, die mit einer Art Gleichgewichtssinn verglichen werden können und fortlaufend Informationen zur Agentenseite schicken können. Erstere sendet die Drehposition und Rotationsgeschwindigkeit der Augen des Agenten, Letztere dessen Bewegungsgeschwindigkeit und -beschleunigung sowie Rotationsgeschwindigkeit und -beschleunigung. *MsgAgentTurn* wird zur VR gesendet und nimmt eine Gradzahl entgegen. Sie gibt Felice die Anweisung, sich um die jeweilige Gradzahl zu drehen (im Uhrzeigersinn bei einem positiven

Wert und gegen den Uhrzeigersinn bei einem negativen Wert). Für die Implementierung dieser Rotation wurde eine Koroutine verwendet um eine kontinuierliche Drehbewegung zu erzeugen [cor]. Diese wird im ersten Fünftel linear beschleunigt, läuft dann mit Maximalgeschwindigkeit ab und bremst im letzten Fünftel wieder linear ab. Zwei weitere neue Nachrichten sind *MsgAgentMoveTo*, die eine Position im Weltkoordinatensystem entgegennimmt und den Agenten (falls möglich) dorthin laufen lässt, sowie *MsgAgentCancelMovement*, die sämtliche Bewegungen (Laufen und Drehen) abbricht.

Um den Weg des Agenten durch die virtuelle Umgebung entsprechend *MsgAgentMoveTo* berechnen zu können, wurde eine A*-Suche implementiert. Dazu wurde das Kinderzimmer zuerst in ein n x n Gitter aufgeteilt, um eine Karte des Raumes zu erhalten. Für jeden der Gitterpunkte, die in regelmäßigem Abstand etwa auf Kniehöhe des Agenten angelegt sind, wurde getestet, ob er innerhalb der Boundingbox irgendeines Objektes in der Szene liegt. Falls ja, befindet sich an diesem Punkt offenbar ein Objekt und er wird als blockiert behandelt. Um eventuellen durch die Engine bedingten Kollisionen vorzubeugen, die entstehen könnten, wenn sich Felice sehr nahe an einem Gegenstand vorbei bewegt, wurden auch alle Nachbarn der blockierten Gitterpunkte als blockiert gekennzeichnet. Damit entstehen später nur Pfade, die ein wenig Abstand zu Hindernissen lassen. Gitterpunkte, die weder innerhalb eine Boundingbox liegen, noch mit einem Gitterpunkt benachbart sind, der das tut, sind frei und vom Agenten begehbar. Die A*-Suche nimmt nun das erstellte Gitter und behandelt jeden seiner Punkte als Knoten. Start- und Zielpunkt werden dem Algorithmus als Raumpunkte (3D-Vektoren) übergeben und auf den am nächsten liegenden Punkt des Gitters abgebildet. Vom Startpunkt ausgehend werden nun jeweils die Nachbarpunkte betrachtet und anhand ihres Abstandes zum diesem und der Manhattan-Distanz zum Zielpunkt gewichtet. Dieses Verfahren wird nun für den vielversprechendsten der so entdeckten Punkte fortgesetzt (wobei als Distanz zum nächsten Punkt jeweils der gesamte bisherige Pfad gesehen wird), bis man am Zielpunkt angelangt ist.

Der Agent kann sich auf 2 verschiedene Weisen entlang des Pfades bewegen. Die erste orientiert sich direkt am Raumgitter: Mittels Koroutinen dreht sich Felice immer zuerst in Richtung des nächsten Pfadknotens und läuft dann so lange in diese Richtung bis sie diese ändern muss. Ihr erstes Ziel ist dabei der zweite Knoten des berechneten Pfades, da der erste in etwa ihrer Position auf dem Gitter entspricht. Drehung und Vorwärtsbewegung laufen dabei analog zu den Nachrichten *MsgAgentTurn* und *MsgAgentMove* ab. Bei der zweiten Art der Bewegung läuft Felice entlang einer Bezier-Kurve durch das Kinderzimmer, deren Kontrollpunkte durch die Gitterpunkte des Pfades gegeben sind. Die aktuelle Position des Agenten auf der Kurve wird dabei mit Hilfe des Algorithmus von de Casteljau approximiert. Dieser basiert auf einer rekursiven Vereinfachung des Kontrollpolygons, bei der nach und nach die Menge der Kontrollpunkte reduziert wird, bis nur noch ein Punkt übrig ist, der sich dann auf der Bezier-Kurve befindet. Durch eine inkrementelle Erhöhung des Eingabewertes (bereits zurückgelegte Strecke im Bezug auf die Gesamtstrecke) erhält man eine Bewegung entlang der Kurve.

In jedem Szenario wurde die beobachtende Kamera so positioniert, dass sowohl Felice als auch die Objekte, mit denen sie interagiert, in der Szenenansicht zu sehen sind. In der ursprünglichen Szenenansicht waren im oberen rechten Bereich die Kamerabilder vom linken und rechten Auge des Agenten nebeneinander angeordnet. Um mehr von der ei-

gentlichen Szene sehen zu können, wurden diese beiden aufeinander geschoben (Bild der linken Augen-Kamera obenauf). Da die Bilder der Augen beinahe identisch sind, hat das keinen Nachteil für den Beobachtenden. Um die Auswertung der verschiedenen Spatial Cognition Szenarios zu erleichtern, wird bei jedem Start eine Textdatei erstellt, die von allen Objekten mit denen Felice interagiert die Positionen sowohl im Gitter- als auch im Weltkoordinatensystem enthält.

4 Spatial Cognition

4.1 Object Localization

Objekterkennung ist die Aufgabe ein gegebenes Zielobjekt in einer Szene zu lokalisieren und zu erkennen. Diese Aufgabe wird täglich vom Menschen ausgeführt, beispielsweise wenn man die Fernbedienung vom Wohnzimmertisch aufheben möchte. Psychologen untersuchen dies am Beispiel der gesteuerten visuellen Suche (engl. ”guided visual search”), in der ein gegebenes Ziel zwischen verschiedenen Distraktoren (ablenkenden Objekten) gesucht werden muss. Wenn das Ziel sich von den Distraktoren in einer Eigenschaft klar unterscheidet, ist es leicht erkennbar (Pop-Out Effekt). Je ähnlicher das Ziel den Distraktoren ist, umso langsamer ist die Suche.

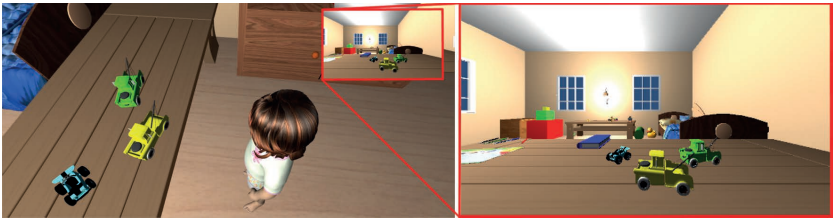


Abbildung 3: Felice steht vor dem Tisch auf dem sich die 3 Spielzeugautos befinden, von denen sie 2 lokalisieren soll.

Das Object Localization Szenario besteht aus zwei Objekten, die angeschaut werden sollen und einem Distraktor. Bei allen dreien handelt es sich um Spielzeugautos, die sich farblich unterscheiden, wobei der Distraktor zusätzlich in seiner Form abweicht. Zu Beginn wird Felice vor einem Tisch platziert und schaut geradeaus über diesen hinweg (Abb. 3). Auf dem Tisch befinden sich direkt vor ihr die drei Spielzeugautos (die zu erkennenden in gelb und grün und der Distraktor in hellblau). An dieser Stelle wird in Zukunft ein Gehirnmodell auf Agentenseite die Objekterkennung durchführen und die beiden zu erkennenden Objekte lokalisieren und anschauen. In der Test-Implementierung dieses Szenarios wird diese Lokalisierung als erfolgreich angesehen und die beiden Objekte werden nacheinander vom Agenten betrachtet.

4.2 Body Displacement

Im täglichen Leben sind wir uns der Objekte in unserem Umfeld nahezu perfekt bewusst [VPM07]. Auch wenn wir uns bewegen, scheinen wir keine Probleme damit zu haben die Übersicht zu behalten. Dieses scheinbar automatische Verhalten der Aktualisierung räumlicher Kontextinformationen funktioniert sogar in völliger Dunkelheit. Aber trotz umfassender Untersuchungen bleibt die Berechnungsgrundlage eines solchen Aktualisierens ein kontrovers diskutiertes Problem. Zentrale Fragen in den kognitiven Neurowissenschaften der räumlichen Erinnerung beziehen sich auf die Referenzkoordinatensysteme, die genutzt werden um Positionen zu repräsentieren. Beispielsweise ob diese egozentrisch (relativ zu Teilen des Körpers) oder allozentrisch (relativ zur externen Umgebung) sind, die Dauer über die verschiedene Repräsentationen beibehalten werden, wozu sie genutzt werden und wie sie miteinander interagieren. Im Szenario "Body Displacementßoll versucht werden, den Antworten auf diese Fragen näher zu kommen. Dem virtuellen Agenten wird vor einer Translation seines gesamten Körpers kurzzeitig ein Zielobjekt eingeblendet. Nach der Verschiebung soll er sich an die Position des Zielobjektes erinnern und es anschauen. Die Fehler liefern dabei Hinweise ob die Körper-Translation in einem augenzentrierten und/oder körperzentrierten Koordinatensystem enkodiert ist.

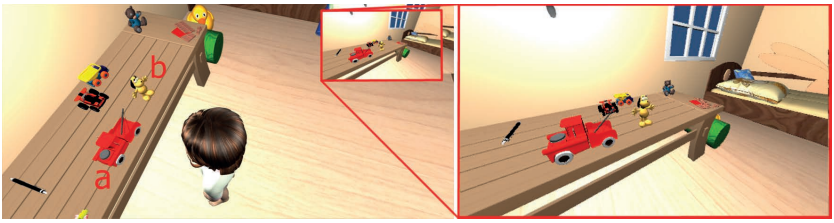


Abbildung 4: Felice fixiert den Hund (b), während das rote Auto, dessen Position sich gemerkt werden soll (a), kurz eingeblendet wird.

Auch hier wird Felice wieder vor einem Tisch platziert, auf dem sich verschiedene Objekte befinden. Ihr Blick geht gerade nach vorne. Die zwei wichtigen Objekte sind hierbei ein kleiner Spielzeughund sowie ein rotes Spielzeugauto. Letzteres wird allerdings nur kurz eingeblendet. Zu Beginn des Szenarios wird der Blick des Agenten auf einen Fixationspunkt (den Hund) zentriert. Im Anschluss wird kurzzeitig das Zielobjekt (rotes Auto) eingeblendet (Abb. 4), das der Agent in seinem peripheren Sichtfeld wahrnimmt, und Felice danach kontinuierlich vor dem Tisch nach rechts verschoben. In der Zeit in der das Zielobjekt sichtbar ist, soll sich dessen Position gemerkt werden. Dieses Merken und das spätere Erinnern an die Objekt-Position wird in Zukunft von einem Gehirnmodell realisiert. In der Test-Implementierung wird wieder davon ausgegangen, dass sich die Position korrekt gemerkt wurde. Die Seitwärts-Verschiebung des Agenten soll eine externe Krafteinwirkung simulieren durch die er zur Seite geschoben wird (beispielsweise einen Stoß durch eine andere Person). Der Blick von Felice ist während der gesamten Bewegung und auch danach weiterhin auf den Fixationspunkt gerichtet. Anschließend wird die erinnerte Position

des Zielobjekts von ihr angeschaut. Als letzter Schritt wird nun das rote Auto noch einmal eingeblendet, um die Korrektheit zu überprüfen.

4.3 Multiple Attention Pointers

Jedes Mal wenn die Augen sich bewegen, um potentiell relevante Objekte in den scharfen Bereich des Sehens zu bringen, muss das visuelle System mit starken Verschiebungen des Bildes auf der Retina umgehen können [Rol11]. Diese retinalen Bildverschiebungen sind nicht bewusst wahrnehmbar. Und noch wichtiger: Wir verlieren nicht die Objekte in der Szene, die von momentanen Interesse sind und eventuell das Ziel zukünftiger Augenbewegungen werden. Die Unfähigkeit Änderungen in unbeaufsichtigten Teilen der Szene zu bemerken, deutet darauf hin, dass Aufmerksamkeit das Verschiebungsproblem nur für eine kleine Anzahl an Positionen löst. Aufmerksamkeitszeiger spezifizieren die Position aktuell beachteter Objekte, ob Ziele folgender Sakkaden oder nicht, und ermöglichen damit sowohl das Planen von Aktionen mit ihnen, als auch die verbesserte Bearbeitung an diesen Positionen. Diese Zeiger auf Ziele von Sakkaden werden durch einen vorhersagbaren Prozess, kurz bevor das Auge anfängt, sich zu bewegen, aktualisiert. Dieser Prozess verschiebt die Aufmerksamkeit, im Gegensatz zur Sakkade, in die Richtung von Positionen, die den aktuellen Zielen weder distal noch proximal entsprechen. Damit wird, bevor die Augenbewegung dort ankommt, die Position vorausgesehen, die die Ziele auf der Retina haben werden, nachdem die Sakkade durchgeführt wurde.

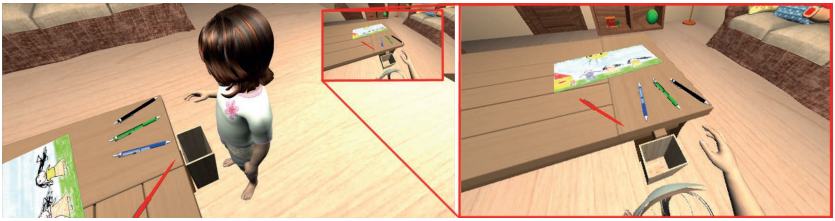


Abbildung 5: Felice startet die Greifbewegung in Richtung des blauen Stiftes.

Im Multiple Attention Pointers Szenario geht es darum mehrere Stifte zu greifen und in eine dafür vorgesehene Box zu legen. Währenddessen wird ein anderer Stift ïm Auge behalten”. Dieser wird dann unvorhergesehen entfernt und Felice soll darauf reagieren. Der Agent wird wie gewohnt in die Ausgangsposition gebracht. Diese ist vor einem Tisch auf dem sich vier verschiedenfarbige Stifte befinden (von links nach rechts: rot, blau, grün, schwarz). Zwischen Felice und dem Tisch auf dem Boden befindet sich eine Sammelbox, in die die gegriffenen Stifte später fallen gelassen werden. Der Blick des Agenten ist anfangs geradeaus über den Tisch hinweg. Nun wird zuerst der Blick auf den blauen Stift gerichtet und dieser dann gegriffen (Abb. 5). Damit die Stifte nach dem Greifen korrekt in der Hand des Agenten liegen, wurden entsprechende lokale Translationen und Rotationen in der Datei *felice_grasp.csv* vermerkt. Nach Abschluss der Greifbewegung hält Felice

den Stift in ihrer Hand vor sich. Da in der VR-Implementierung bisher keine Animation existiert um gegriffene Objekte an einer gewünschten Stelle wieder abzulegen, besteht die einzige Möglichkeit, den Stift in ihrer Hand loszuwerden, darin, ihn fallen zu lassen. Das ist auch der Grund weshalb sich die Sammelbox auf dem Boden unter der Hand des Agenten befindet und nicht beispielsweise hinter den Stiften auf dem Tisch. Damit die Stifte in der Box liegen bleiben wurden sowohl sie als auch die Box mit entsprechenden Collidern ausgestattet. Nachdem der blaue Stift in die Box fallen gelassen wurde, geht es mit dem zweiten (grünen) Stift weiter. Auch dieser wird wieder, danach gegriffen und schließlich in die Sammelbox fallen gelassen. Als dritter und letzter ist nun der schwarze Stift an de Reihe. Der Agenten richtet erneut seinen Blick auf ihn. Bevor der schwarze Stift allerdings gegriffen werden kann, verschwindet im peripheren Sichtfeld der rote Stift. Dieses einfache Verschwinden soll darstellen, dass beispielsweise eine andere Person den Stift weg nimmt. Am Schluss des Szenarios betrachtet Felice die ehemalige Position des roten Stiftes.

4.4 Vision and Memory

Wir sind oft mit der anspruchsvollen Aufgabe konfrontiert, zu entscheiden, wie wir uns im Angesicht unvollständiger sensorischer Informationen verhalten [BBB07]. Um solche Aufgaben zu lösen, müssen wir uns auf interne Repräsentationen von den Positionen der Objekte in ihrer Umwelt verlassen. Zu verstehen wie eine höhere kognitive Funktion wie die Erinnerung aus der Zusammenarbeit von Neuronen und Synapsen in unserem Gehirn entsteht ist eine der faszinierendsten Herausforderungen kognitiver Neurowissenschaften. Ein solches Verständnis würde als Brücke zwischen den neurophysiologischen und verhaltensbasierten Beschreibungsebenen der Neurowissenschaften dienen und es erlauben, die Lösung von Fragen von einem Beschreibungslevel durch Rückschlüsse auf ein anderes zu bestärken. Weiterhin könnte eine Theorie, die Zell- und Verhaltensebene verknüpft, zur Entwicklung von experimentellen Vorhersagen von einer Ebene auf eine andere führen und die Fähigkeit, Verhaltenssymptome ihrer zugrundeliegenden Symptomatik zuzuordnen, verbessern. Was das Entwickeln solch eines Verstehens der Erinnerung angeht, stellt die räumliche Erinnerung einen guten Startpunkt bereit, da man ähnliche Paradigmen bei Menschen und anderen Tieren benutzen kann.

Abbildung 6: Felice hat sich die Position der Gummibärchen (a) eingeprägt und begibt sich zum Zimmerausgang.

Das Vision and Memory Szenario besteht aus der Aufgabe ein Objekt anzuschauen, sich dessen Position im Raum zu merken, den Raum dann zu verlassen und schließlich den Raum erneut zu betreten und den Weg zum erinnerten Objekt wiederzufinden. Der Ablauf wurde zweigeteilt und die beiden Teile können auch unabhängig voneinander durchgeführt werden. Im ersten Teil wird Felice vor einen Tisch gestellt und blickt geradeaus an die Wand dahinter. Vor ihr auf dem Tisch befinden sich einige rote und gelbe Gummibärchen. Diese werden angeschaut. Dann wird der Blick des Agenten wieder in die Ausgangsposition versetzt und ihm wird der Befehl gegeben, durch das Zimmer in Richtung Tür zu laufen (Abb. 6). Ziel ist eine Position, die kurz vor der Tür liegt. Dabei steht der Tisch in der Mitte des Raumes zwischen Felice und dem Ziel, so dass sie einen Bogen laufen muss. Der zweite Teil startet vor der Tür mit Blickrichtung ins Zimmer. Hier soll sich nun in Zukunft ein Gehirnmodell an die Position der Gummibärchen erinnern bzw. diese suchen. In der Test-Implementierung wird angenommen, dass die Position bekannt ist und Felice läuft dorthin. Am Ende des Szenarios steht sie nun wieder vor den Gummibärchen und schaut sie erneut an.

5 Fazit und Ausblick

Das Ziel dieser Arbeit wurde erreicht: Es liegt nun eine stabil laufende, virtuelle Versuchsumgebung vor, die vier verschiedene Szenarien bereitstellt um Objekterkennung, Raumorientierung und visuelle Aufmerksamkeit auf einem oder mehreren Objekte untersuchen zu können. Die Implementierungen bieten die Möglichkeit, kleinere Änderungen ohne großen Aufwand vorzunehmen, den Ablauf anzupassen und die Szenarien auf andere Problemstellungen im visuellen Bereich zu erweitern. Die Farbe, Position und Anzahl der Objekte kann jederzeit verändert werden und auch das Nutzen anderer 3D-Modelle um die Form zu variieren stellt kein Problem dar. Natürlich gibt es noch viele Möglichkeiten, die virtuelle Umgebung in Zukunft je nach Bedarf auszubauen, wofür die momentanen Implementierungen eine hervorragende Grundlage bieten. Dem Agenten könnten noch mehr Interaktionsmöglichkeiten mit seiner Umgebung (und die zugehörigen Animationen) gegeben werden. Beispiele dafür sind unter anderem ein Seitwärtsschritt für das Body Displacement Szenario, eine Animation um gegriffene Objekte an einer bestimmten Position abzulegen und die Möglichkeit Türen zu öffnen und damit tatsächlich das Zimmer zu verlassen. Auch in grafischer Hinsicht ist noch eine Menge möglich (Raytracing, Motion Blur, bessere Materialien etc.). Dabei muss man allerdings abwiegen, ob man eine maximal realistische Szenerie benötigt, oder ob es in Hinsicht auf die experimentellen Tests nicht sogar von Vorteil ist, die Darstellung möglichst einfach zu halten. Ebenfalls denkbar ist eine zukünftige Umsetzung in der Robotik, die sich an der VR orientiert.

Literatur

[ann] `http://annarchy.readthedocs.org/en/latest/` 14.08.2015.

[Ant14] M. et al. Antonelli. A Hierachical System for a Distributed Representation of the Perpersonal Space of a Humanoid Robot. *IEEE Transactions On Autonomous Mental Development*, 6(4):259–273, 2014.

[BBB07] P. Byrne, S. Becker und N. Burgress. Remembering the Past and Imagining the Future: A Neural Model of Spatial Memory and Imagery. *Psychological Review*, 114(2):340–375, 2007.

[BH08] F. Beuth und F. H. Hamker. The guidance of vision while learning categories. *Poster on the Bernstein Conference on Computational Neuroscience (BCCN)*, 2008.

[CA05] P. Cavanagh und G. A. Alvarez. Tracking multiple targets with multifocal attention. *Trends in Cognitive Sciences*, 9(7):349–354, 2005.

[cor] `http://docs.unity3d.com/ScriptReference/MonoBehaviour.StartCoroutine.html` 15.10.2015.

[Ham05] F. H. Hamker. The emergence of attention by population-based inference and its role in distributed processing and cognitive control of vision. *Computer Vision and Image Understanding*, 100:64–106, 2005.

[pro] `https://developers.google.com/protocol-buffers/docs/overview` 20.09.2015.

[Rol11] M. et al. Rolfs. Predictive remapping of attention across eye movements. *Nature Neuroscience Advance Online Publication*, 2011.

[spa] `http://www.spacecog.eu/project/index.html` 14.08.2015.

[VPM07] S. Van Pelt und W. P. Medendorp. Gaze-Centered Updating of Remembered Visual Space During Active Whole-Body Translations. *J Neurophysiol 97*, Seite 1209–1220, 2007.

[VWVO88] S. M. C. I. Van Wetter und A. J. Van Opstal. Experimental test of visuomotor updating models that explain perisaccadic mislocalization. *Journal of Vision*, 8(14):1–22, 1988.

Untersuchungen und Optimierung der Kontexterkennung auf mobilen Endgeräten mittels Sensorfusion

Maik Benndorf

Hochschule Mittweida, Fakultät Angewandte Computer- und Biowissenschaften,
Technikumplatz 17, 09648 Mittweida
benndorf@hs-mittweida.de

Zusammenfassung. Kontextsensitive Anwendungen sollen die Interaktion zwischen Mensch und Maschine verbessern. Im einfachsten Fall bedeutet dies, dass die Anwendung auf die Ausrichtung des Gerätes mit einer Anpassung der Darstellung (Hochformat / Querformat) reagiert. Dabei sind diese Anwendungen besonders für die mobilen Geräte geeignet, da sich die Umgebungsbedingungen hier häufig ändern können. Informationen über die Umgebungsbedingungen können z. B. aus den eingebauten Sensoren gewonnen werden. So können mit einzelnen Sensoren bereits zahlreiche einfache Situationen (z. B. Gehen, Rennen) erkannt werden. Ziel dieses Vorhabens ist es, komplementäre Sensoren zu fusionieren, um so komplexere Situationen zu erkennen.

1 Motivation

In vielen Situationen passen Menschen ihr Verhalten an äußere Umgebungsbedingungen an. Im Gegensatz dazu reagieren die meisten Computer und Programme nur auf die Eingaben des Benutzers. Sie liefern unabhängig von äußeren Einflüssen immer das gleiche Ergebnis. Mit kontextsensitiven Anwendungen soll die Interaktion zwischen Mensch und Computern verbessert werden. Dabei werden zur Bestimmung des Ergebnisses weitere Informationen wie z. B. die Ausrichtung des Gerätes herangezogen. Für den Begriff Kontext liegen bis heute zahlreiche [1–3], jedoch keine eindeutige Definition vor. Für diese Arbeit wird die Definition von Abowd u. a. [4] gewählt:

> *Kontext ist jede Art von Information, die zur Charakterisierung einer Situation einer Entität genutzt werden kann. Eine Entität ist eine Person, ein Platz oder ein Objekt, das als relevant für die Interaktion zwischen einer Anwendung und dem Nutzer angesehen werden kann. Dazu zählen auch der Nutzer und die Anwendung selbst.*

Kontextsensitive Anwendungen sind besonders für mobile Geräte geeignet, da sich die Umgebungsbedingungen für eine Anwendung hier häufig ändern können. Zu diesen Geräten gehören z. B. moderne Mobiltelefone und Smartphones. Diese Geräte sind mit verschiedenen Sensoren (z. B. Lage-, Orientierungs- und

Beschleunigungssensoren) ausgestattet, deren Messdaten zur Bestimmung des Kontextes herangezogen werden können. Daneben bieten z. B. die Verfügbarkeit von WLAN-Netzwerken oder die Signalstärken des Funkmoduls weitere Informationen für die Kontexterkennung.

Zur Erkennung des Kontextes werden bisher oftmals nur einzelne Sensoren oder einfache statische Verbünde von Sensoren (z. B. Beschleunigungssensor in Verbindung mit dem GPS-Modul) eingesetzt. In der Literatur wird dies begründet mit den knappen Ressourcen dieser Geräte. Dazu zählen unter anderen der Energieverbrauch, die Rechenleistung und der begrenzte Speicher. Auch aus diesem Grund werden in der Regel Client-Server-Architekturen zur Bestimmung des Kontextes eingesetzt.

Forschungsbedarf besteht unter anderem darin, komplexe Kontexte oder Aktivitäten zu erkennen. Als Aktivität wird die aktuelle Tätigkeit des Nutzers bezeichnet. Bisher können zahlreiche einfache Aktivitäten (z. B. Gehen, Laufen oder Zähne putzen) hinreichend zuverlässig erkannt werden. Herausforderung bieten unter anderen kombinierte oder konkurrierende Tätigkeiten. Eine kombinierte Aktivität besteht aus mehreren einzelnen Tätigkeiten, so sind z. B. das Stehen, Gehen, Laufen und Sitzen Bestandteile der komplexen Aktivität Tennisspielen. Als konkurrierend werden sich überlagernde Aktivitäten bezeichnet. Dazu zählt z. B. das Gehen während des Zähneputzens. Weiterer Forschungsbedarf besteht im Erkennen zukünftiger Aktivitäten, wie dem Mittagessen (eine Anwendung könnte in diesem Fall ein Restaurant empfehlen). Neben den Aktivitäten besteht der Forschungsbedarf im Gewinn weiterer Informationen wie z. B. dem Ableiten des Geschlechts, des Alters oder der Motivation des Nutzers.

2 Vorarbeiten

Die Kontexterkennung ist ein sehr aktives Forschungsgebiet mit vielen Teilthemen. In diesem Abschnitt wird der aktuelle Stand in Wissenschaft und Technik für die einzelnen Teilgebiete beleuchtet, welche auch das Vorhaben berühren.

2.1 Komplexe Aktivitäten

Zur Erkennung komplexer Aktivitäten gibt es verschiedene Ansätze. Die meisten Arbeiten verwenden ein einfaches Verfahren mit nur einem Sensor. Dabei kommt oftmals der Beschleunigungssensor zum Einsatz. Dieser Sensor ist auf vielen Geräten verfügbar und zeichnet sich durch seinen geringen Energieverbrauch aus. Die erfassten Sensordaten werden in vielen Fällen in einem einstufigen Verfahren zur Klassifizierung herangezogen. Einen anderen Weg gehen Yan u. a. Sie stellen in ihrer Arbeit [5] einen Ansatz aus zwei Schritten vor. Der erste Schritt ist dabei analog zum einstufigen Verfahren. Im Anschluss daran werden die erkannten Aktivitäten zusammengesetzt, um anschließend statistisch zu ermitteln, um welche komplexe Aktivität es sich handelt. In einem Langzeittest konnte so

eine durchschnittliche Genauigkeit von ca. 77 % erreicht werden. In ihrem Ausblick verweisen die Autoren darauf, dass zukünftig weitere Sensoren einbezogen werden sollten.

2.2 Dynamische Sensoren

Das dynamische Hinzufügen bzw. Entfernen von Sensoren kann genutzt werden, um den Energieverbrauch zu senken. Das bedeutet, dass Sensoren erst zugeschaltet werden, wenn sie zur Erkennung benötigt werden. Ein aktiviertes GPS-Modul ist z. B. nicht sinnvoll, wenn sich die Position des Telefons nicht ändert (z. B. während des Schlafs). Verschiedene Arbeiten nutzen Verbünde zur Aktivitätserkennung. Ryder u. a. verwenden in ihrer Arbeit [8] den Beschleunigungssensor und das GPS-Modul. Dabei wird das GPS-Modul abgeschaltet, sobald der Nutzer ein Gebäude betritt. Wang u. a. [9] nutzen zur Erkennung der Aktivitäten den Beschleunigungssensor, GPS, einen WiFi-Scanner und das Mikrofon. Dabei werden diese Sensoren für ausgewählte, einfache Aktivitäten anhand eines zuvor definierten hierarchischen Musters bei Bedarf zu- bzw. abgeschaltet. Der Forschungsbedarf besteht laut den Arbeiten darin, weitere Sensoren und Aktivitäten einzubeziehen und das Zu- bzw. Abschalten der Sensoren dynamischer zu gestalten.

2.3 Klassifikation auf dem mobilen Gerät

Einige Arbeiten [6, 7, 9] führen die Klassifikation auf dem mobilen Gerät durch. Dazu kommen nur wenige ausgewählte Algorithmen zum Einsatz (z. B. Entscheidungsbäume, SVM, naive Bayes oder der KNN-Algorithmus). Komplexe oder mehrstufige Algorithmen werden in der Regel nur in Verbindung mit einer Client-Server-Architektur eingesetzt, da die zuvor genannten Limitierungen hier nicht existieren.

3 Zielsetzung und Abgrenzung

Das Ziel dieses Vorhabens sind Optimierungen im Bereich der Kontexterkennung. Dazu sollen die genannten Problemstellungen aufgegriffen werden. Hierzu zählen die komplexeren Aktivitäten, das dynamische Zu- bzw. Abschalten von Sensoren und die Klassifizierung auf dem mobilen Gerät.

3.1 Die Arbeitsschritte des Vorhabens

In diesem Abschnitt werden die einzelnen Teilschritte des Vorhabens benannt. Dazu wurde das Vorhaben in sieben Phasen eingeteilt. Ein zeitlicher Ablaufplan kann dem folgenden Gantt-Diagramm entnommen werden.

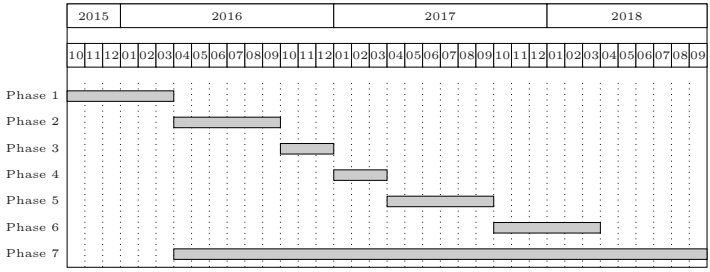


Abb. 1. Arbeitsschritte des Promotionsvorhabens

Phase 1 - Vorbereitung und Literaturrecherche Zu Beginn der Arbeit steht eine Literaturrecherche, mit welcher der aktuelle Stand aus Wissenschaft und Technik in Bezug auf die Kontexterkennung, die verwendeten Sensoren und die Klassifizierung auf dem mobilen Gerät näher beleuchtet werden soll.

Phase 2 - Evaluierung der bisher eingesetzten Sensoren Das Ziel dieser Phase ist die Evaluierung der Sensoren, die bisher zur Kontexterkennung eingesetzt wurden. Dazu sollen die recherchierten Sensoren in Bezug auf ihren Beitrag zur Erkennung und ihren Energieverbrauch hin untersucht werden.

Phase 3 - Untersuchung bisher nicht betrachteter Informationsquellen Neben den bisher eingesetzten Sensoren sollen in dieser Phase weitere lokale Sensoren einbezogen werden. Dazu zählen z. B. die Near Field Communication und die ANT+-Technologie. Hierzu sind weitere Recherchen und ein Abgleich mit dem derzeitigen Stand der Technik notwendig.

Phase 4 - Integration externer Sensoren Nachdem bisher nur interne Sensoren betrachtet wurden, sollen in dieser Phase externe Sensoren einbezogen werden. Dazu zählen neben den Wearables auch die Sensoren benachbarter Geräte. Für den Datenaustausch zwischen den Geräten sollen Aggregations- bzw. Kompressionsverfahren untersucht werden.

Phase 5 - Sensorfusion Zu diesem Zeitpunkt wurden die verschiedenen Sensoren einzeln untersucht. In dieser Phase sollen Verbünde einzelner Sensoren evaluiert werden. Im Anschluss daran sollen Methoden geschaffen werden, mit denen sich die Sensoren dynamisch hinzufügen bzw. entfernen lassen. Eine Möglichkeit dazu bietet der hierarchische Ansatz von Wang u. a. [9].

Phase 6 - Klassifizierung auf dem mobilen Gerät Bisher wird zur Kontexterkennung in vielen Fällen eine Client-Server-Architektur eingesetzt. Da auf

einem Server mehr Rechenleistung und kaum Limitierungen in Bezug auf Energie und Speicher bestehen, können bei diesem Konzept komplexe Algorithmen und Modelle eingesetzt werden. Moderne Smartphones unterliegen einer ständigen Weiterentwicklung und viele dieser Geräte sind mit leistungsfähigen Mehrkernprozessoren ausgestattet. Daneben bietet die GPU dieser Geräte Möglichkeiten zur Berechnung komplexer Algorithmen. In dieser Phase soll zunächst der Tradeoff in Bezug auf den Energieverbrauch zwischen einer lokalen Berechnung und der Kommunikation durch die Client-Server-Architektur untersucht werden. In Abhängigkeit von dieser Untersuchung werden dann weitere Methoden zur lokalen Berechnung untersucht.

Phase 7 - Dissertation Die Phase 7 startet im Anschluss an die Literaturrecherche und verläuft von diesem Moment an parallel zu den weiteren Phasen. Dazu werden die Ergebnisse der einzelnen Phasen in Bezug auf die Forschungsfragen evaluiert und in die Dissertation übernommen.

3.2 Nachhaltigkeit des Themas

Die Automatisierung von Abläufen im täglichen Leben nimmt immer weiter zu. Mit den kontextsensitiven Anwendungen wurde ein weiter Schritt in diese Richtung unternommen. Dem Nutzer wird der Umgang mit einem technischen Gerät weiter erleichtert. Dabei muss es sich bei dem technischen Gerät nicht zwangsläufig um das Smartphone handeln. Denkbar ist es auch, dass dieses Konzept z. B. auf Smartwatches oder Datenbrillen übertragen werden kann. Ein Vorteil dieses Forschungsbereichs ist es, dass dieser auf eine Vielzahl von Geschäftsfeldern angewendet werden kann. So können neben der wissenschaftlichen Gemeinschaft weitere Branchen als potenzielle Nachfrager der Forschungsergebnisse angesehen werden. Dazu gehört die Gesundheitsbranche, in der mit kontextsensitiven Anwendungen bereits heute die Heilung der Patienten überwacht werden kann. Daneben kann auch die Quantified-Self-Bewegung und schließlich die Branche der Softwareentwicklung als Nachfrager angesehen werden.

Danksagung

Bei diesem Vorhaben handelt es sich um eine kooperative Promotion zwischen den Fakultäten Angewandte Computer- und Biowissenschaften der HS Mittweida und Informatik der TU Chemnitz. Betreut wird dieses Vorhaben durch Professor Dr. habil. Thomas Haenselmann (Inhaber der Professur Algorithmen, Computergrafik, Bildverarbeitung an der HS Mittweida) und Jun.-Prof. Dr. Marc Ritter (Inhaber der Professur Media Computing an der TU Chemnitz). Finanziert wird dieses Vorhaben im Rahmen des Forschungsprojekts "FloodEvac - Verletzbarkeit von Transport-Infrastrukturen - Warnung und Evakuierung bei Hochwasser"durch das BMBF (Bundesministerium für Bildung und Forschung, FKZ 13N13196).

Literatur

1. Schilit, B. N., Adams, N., Want, R.: Context-aware computing applications. Mobile Computing Systems and Applications. 85–90 (1994)
2. Schilit, B. N.: A system architecture for context-aware mobile computing (1995)
3. Schmidt, A., Beigl, M., Gellersen, H.: There is more to context than location. Computers & Graphics 23/6 893–901 (1999)
4. Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., Steggles, P.: Towards a better understanding of context and context-awareness. Handheld and ubiquitous computing 304–307 (1999)
5. Yan, Z., Chakraborty, D., Misra, A., Jeung, H., Aberer, K.: Semantic activity classification using locomotive signatures from mobile phones. No. EPFL-REPORT-174016 (2012)
6. Kose, M., Incel, O. D., Ersoy, C.: Online human activity recognition on smart phones. Workshop on Mobile Sensing: From Smartphones and Wearables to Big Data 11–15 (2012)
7. Reddy, S., Mun, M., Burke, J., Estrin, D., Hansen, M., Srivastava, M.: Using mobile phones to determine transportation modes. Transactions on Sensor Networks (TOSN) 6/2 13 (2010)
8. Ryder, J., Longstaff, B., Reddy, S., Estrin, D.: Ambulation: a tool for monitoring mobility patterns over time using mobile phones. 12th International Conference on Computational Science and Engineering 4 927–931 (2009)
9. Wang, Y., Lin, J., Annavaram, M., Jacobson, Q., Hong, J. I., Krishnamachari, B., Sadeh, N.: A framework of energy efficient mobile sensing for automatic user state recognition. 7th international conference on Mobile systems, applications, and services 179–192 (2009)

Schallquellenlokalisierung zur Sprecheridentifizierung für Multi-User Dialogschnittstellen

René Schmidt * and Wolfram Hardt

Technische Universität Chemnitz
Fakultät für Informatik
Professur Technische Informatik
{rene.schmidt,wolfram.hardt}@informatik.tu-chemnitz.de

Zusammenfassung Sprachbasierte Dialogschnittstellen stellen die natürlichste Form der Interaktion zwischen Mensch und Maschine dar und haben sich bereits im Feld der Museumsroboter etabliert. So verwenden Roboter wie Fritz oder Robotinho sprachbasierte Dialogsysteme um Besuchern Ausstellungsstücke zu erklären. Hierbei handelt es sich meist um rudimentäre Dialogsysteme, welche nur für einen Anwender zugeschnitten sind. In diesem Paper wird ein Algorithmus zur Lokalisierung von Sprechern vorgestellt, welcher die Grundlage für ein Multi-User Dialog-Interface zur Nutzung auf mobilen Robotern bietet. Dabei wird, neben der hohen Genauigkeit der Lokalisierung, besonderer Fokus auf eine hohe Berechenbarkeit der Laufzeit sowie eine möglichst gute Parallelisierbarkeit der Berechnung gelegt.

1 Einleitung

Museumsroboter wie Robotinho [2], Fritz [1] oder Minerva [7] haben sich im Museumsumfeld bereits gut etabliert. Als Kommunikationsschnittstelle wird häufig ein Touch-Interface verwendet. Doch das intuitivste Kommunikationsmedium ist seit jeher die menschliche Sprache. Einige Roboter wie FRITZ verwenden bereits sprachbasierte Dialog-Interfaces zur Kommunikation mit Besuchern [1]. Diese sind jedoch in der Regel rudimentäre Dialogsysteme, welche versuchen intuitive Dialoge mit Füllwörtern zu erreichen, wenn die Berechnungszeiten zu lange dauern [5]. Psychologische Erkenntnisse zeigen jedoch, dass eine angemessene Wartezeit zwischen den Zeitpunkten der Fragestellung und Antwortgabe je nach Kontext variiert und angepasst werden muss [6]. Um dieses Ziel zu erreichen ist es notwendig ein komplett berechenbares Dialogsystem zu realisieren, mit der Zielstellung die Berechnungszeit möglichst minimal zu halten und einen möglichst hohen Grad an Berechenbarkeit zu schaffen. Aus diesem Grund sind dezentrale Ansätze nicht möglich. Im Hinblick der Nutzung des Systems auf einem mobilen Roboter ist somit eine spezialisierte Hardware unumgänglich. Bisherige Lösungsansätze würden eine digitale Vorverarbeitung der Signale in

* The author gratefully acknowledges funding by the DFG (GRK 1780/1).

einem FPGA oder ASIC und einer anschließenden Weiterverarbeitung in einem Prozessor nahelegen. Hierbei hat man jedoch sehr begrenzte Möglichkeiten der Hardware/Software Partitionierung, da die Übertragungszeiten zwischen Prozessor und Hardware sehr hoch sind. Hier bietet die neue SoC Architektur von Xillinx und Altera ganz neue Möglichkeiten der Hardware/Software Partitionierung durch die gemeinsame Nutzung von AXI-Interfaces. Dies ermöglicht einen variablen Austausch zwischen FPGA und Prozessor, mit dem Ziel die Parallelisierungseigenschaften des FPGAs optimal zu nutzen. Somit kann eine exakte Berechenbarkeit durch eine optimierte Hardware/Software Partitionierung gewonnen werden. Durch die Nutzung von Standalone-Anwendungen gibt es keine durch Interrupt begründeten Verzögerungen der Berechnungen, wie dies bei der Nutzung von Betriebssystemen der Fall wäre. Auf diese Weise wird ein deterministischer Ablauf garantiert. Eine weitere Herausforderung des Museumskontextes bildet die Notwendigkeit zur gleichzeitigen Interaktion mit mehreren Personen. Daher ist die Nutzung von Multi-User Anwendungen in derartigen Umgebungen unumgänglich. Um die jeweiligen Sprecher zu unterscheiden, müssen diese aus den Mikrophonsignalen extrahiert werden, wie in [4, 9] gezeigt. Anschließend werden die Sprechersignale über LPCC Feature einander zugeordnet. Um eine Person im Nachhinein zu verfolgen, ist dieses Verfahren jedoch zu ungenau [10]. Um dieses Problem zu lösen ist eine Lokalisierung notwendig, welche den primären Fokus dieses Papers bildet. Standardmäßige Lokalisierungsmechanismen berechnen die Position eines Sprechers unter der Angabe eines Winkels und der Distanz [3, 8]. Dies ist jedoch ebenfalls ungeeignet um Personen zu verfolgen. Hierfür bietet sich eine Lokalisierung im dreidimensionalen Raum besser an. Im Folgenden wird daher ein Algorithmus zur Lokalisierung eines Sprechers im dreidimensionalen Raum vorgestellt, welcher sich sehr gut parallelisieren lässt und somit sehr gut für die Implementierung auf einem FPGA geeignet ist. Anschließend werden relevante Ergebnisse der Simulation vorgestellt und diskutiert.

2 Algorithmus

Das Ziel des Algorithmus besteht darin, einen Sprecher im dreidimensionalen Raum zu lokalisieren. Dieses Problem kann wie in Abbildung 1 dargestellt werden. Der Algorithmus basiert auf dem gängigen TDOA Ansatz [3, 8]. Das Hauptproblem ist hierbei jedoch die ungenügende Genauigkeit des Verfahrens bei geringen Abtastraten [8]. Bei TDOA Verfahren wird die Differenz von Signalen mit Hilfe einer Kreuzkorrelation bestimmt und anhand dieser Ergebnisse der Winkel zwischen den beiden Signalen α_{xy} berechnet. Bei der Nutzung von drei Mikrophonen kann mit Hilfe einer Triangulation die Entfernung zur Schallquelle d bestimmt werden. Für die folgenden Betrachtungen gelten die Definitionen für Mikrophone M_m und Schallquelle S.

$$S = \begin{pmatrix} S_x \\ S_y \\ S_z \end{pmatrix} \qquad M_m = \begin{pmatrix} M_{m_x} \\ M_{m_y} \\ M_{m_z} \end{pmatrix} \tag{1}$$

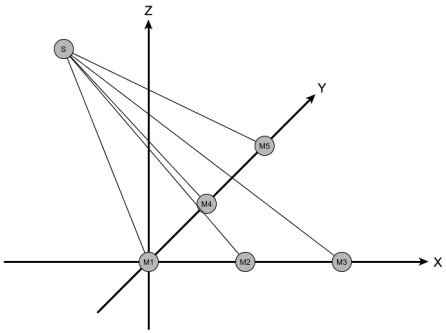


Abbildung 1: Problembeschreibung

Um eine Lokalisierung im dreidimensionalen Raum zu erhalten, werden mindestens vier Mikrophone benötigt. Bei dem zu bestimmenden Gleichungssystem entstehen lediglich partiell lösbare Funktionen, was auf Wurzelbildungen zurück zuführen ist und keine generelle Lösbarkeit garantiert. Aus diesem Grund werden hier fünf Mikrophone verwendet. Die Mikrophone werden wie in Abbildung 1 angeordnet und haben alle einen Basisabstand von 10 cm. Mit Hilfe der Mikrophone M_1 und M_2 wird der Eingangswinkel α_{12} bestimmt. Ein analoges Vorgehen wird für die Mikrophone M_2 und M_3 zur Bestimmung des Winkels α_{23} angewandt, vergleiche Formel 2.

$$\alpha_{xy} = \arcsin(\frac{\Delta_{xy}}{|M_x - M_y|}) \tag{2}$$

$$d = \frac{\Delta_{13}}{2 \cdot (\frac{\sin(\alpha_{12})}{\sin(\alpha_{23})} - 1)} \tag{3}$$

Die so gewonnene Entfernung d bildet die Grundlage zur eindeutigen Lösung der Gleichungssysteme 4 und 5. Da die Lösung im dreidimensionalen Raum zwar möglich ist, jedoch nur zu einem partiell lösbaren Gleichungssystem führt, wird das Problem auf zwei Dimensionen reduziert. Zu diesem Zweck wird das Problem aus Abbildung 1 auf die XY Ebene projiziert und anschließend der Schnittpunkt der beiden Geraden in der XY Ebene bestimmt. Dies findet sich in Gleichungssystem 4 wieder. Weiterhin wird eine Projektion auf die YZ Ebene vorgenommen, was in Gleichungssystem 5 resultiert. Die hier vorgenommenen Projektionen sind nur möglich, wenn die jeweiligen verwendetetn Mikrophone auf einer Ebene liegen.

$$\begin{aligned} (d + \Delta_{12})^2 &= (M_{2_x} - S_x)^2 + (M_{2_y} - S_y)^2 \\ (d + \Delta_{13})^2 &= (M_{3_x} - S_x)^2 + (M_{3_y} - S_y)^2 \end{aligned} \tag{4}$$

$$\begin{aligned} (d + \Delta_{24})^2 &= (M_{4_y} - S_y)^2 + (M_{4_z} - S_z)^2 \\ (d + \Delta_{25})^2 &= (M_{5_y} - S_y)^2 + (M_{5_z} - S_z)^2 \end{aligned} \tag{5}$$

Die Lösung des Gleichungssystem ergibt schließlich die Berechnung der Position der Schallquelle S.

3 Simulationsergebnisse

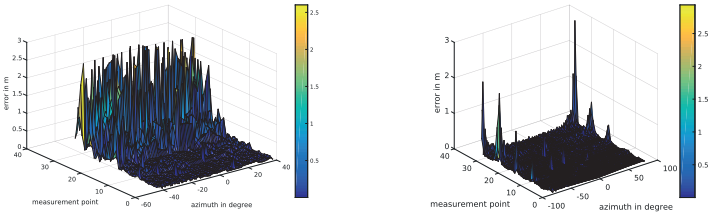
Ziel der Simulation ist es, die theoretischen Grenzen des vorgestellten Algorithmus zu identifizieren und die für die Anwendung notwendigen Minimalanforderungen herauszufinden. Hierfür wurde der Algorithmus in MATLAB implementiert. TDOA Verfahren können hauptsächlich durch zwei Parameter beeinflusst werden, den Basisabstand der Mikrophone untereinander sowie die Abtastrate. Aufgrund des sehr begrenzten Platzes auf einem mobilen Roboter entfällt eine Evaluierung des Mindestabstands der Mikrophone und wurde für die Tests auf 10 cm festgelegt. Um den Einfluss der Abtastrate auf die Genauigkeit zu simulieren, wurden die Eingabeparameter Δ_{xy} für den Algorithmus nach der Formel 6 bestimmt.

$$\begin{aligned} t_m &= \frac{||S - M_m||}{c} \cdot f \\ \delta_{xy} &= \left\lfloor t_x - t_y \right\rfloor \\ \Delta_{xy} &= \frac{\delta_{xy} \cdot c}{f} \end{aligned} \tag{6}$$

S definiert dabei die dreidimensionale Koordinate im Raum und repräsentiert die aktuell zu simulierende Schallquelle. Die Schallgeschwindigkeit wird durch c symbolisiert und die Abtastrate mit f. Um eine möglichst gute Abdeckung des Raumes zu erhalten, wurden um den Ursprung des Koordinatensystem der Mikrophone 5760 Messpunkte simuliert, welche kugelförmig um den Koordinatenursprung angeordnet sind. Dabei wurden ebenfalls unterschiedliche Radien berücksichtigt. Zur automatisierten Berechnung der Simulationspunkte wurde die Kreisbeschreibung aus Formel 7 verwendet, wo n die Gesamtanzahl an Messpunkten auf einem Radius je Winkel ϕ und Radius r bezeichnet und x den aktuellen Messpunkt identifiziert.

$$\begin{aligned} r &\in \{0.5, 1, 2, 3\} \\ \phi &\in \{0, 25, 50, 75, 100, 125, 150, 175\} \\ \theta(x) &= -90 + 180/n \cdot x \\ S_x &= r \cdot \sin(\theta(x)) \cdot \cos(\phi) \\ S_y &= r \cdot \sin(\theta(x)) \cdot \sin(\phi) \\ S_z &= r \cdot \cos(\theta(x)) \end{aligned} \tag{7}$$

Diese Simulation wurde für mehrere Abtastraten durchgeführt. Für jede berechnete Schallquelle wurde der Fehler durch die euklidische Distanz bestimmt.



(a) 192 kHz Abtastrate und 80° Öffnungswinkel

(b) 5 MHz Abtastrate und 164° Öffnungswinkel

Abbildung 2: Simulationergebnisse

Abbildung 2(a) zeigt die Simulationsergebnisse für eine Abtastrate von 192 kHz, welches aktuellen Abtastraten gängiger Audiointerfaces entspricht. Hier lässt sich eine starke Verschlechterung der Genauigkeit mit steigender Entfernung beobachten. Die Werte außerhalb des hier gezeigten Öffnungswinkels sind um ein Vielfaches schlechter und würden die Darstellbarkeit extrem verfälschen. Dieser Effekt kann in Abbildung 2(b) in abgeschwächter Form nachvollzogen werden und resultiert aus dem TDOA Verfahren, da eine Bestimmung des Winkels α_{xy} in den Randgebieten des Öffnungswinkels sehr ungenau ist.

Die folgende Tabelle zeigt den Zusammenhang von Abtastrate, Öffnungswinkel, durchschnittlichem Fehler innerhalb des Öffnungswinkels und der zugehörigen Standardabweichung.

Abtastrate in kHz	Öffnungswinkel in °	Fehler (M) in m	Fehler (SD) in m
192	80	0.3153	0.4900
500	120	0.1229	0.1787
1000	120	0.0680	0.0880
2000	120	0.0361	0.0441
5000	140	0.0256	0.0305
5000	164	0.0343	0.0880

Tabelle 1: Zusammenhang Abtastrate, Öffnungswinkel und Fehler

Die Ergebnisse zeigen, dass eine leichte Erhöhung der Abtastrate einen wesentlich größeren Öffnungswinkel bei besserer Genauigkeit und geringerer Streuung zulässt. Um einen Sprecher in einer Menschengruppe zu identifizieren genügt jedoch keine Genauigkeit von 30 cm, da sich in diesem Bereich bereits Personen überlagern können. Gleichermaßen darf der Öffnungswinkel nicht zu gering sein, andernfalls wird ein zu kleiner Bereich abgedeckt.

4 Zusammenfassung und Ausblick

In diesem Paper wurde ein Lokalisationsalgorithmus vorgestellt, welcher bei angemessener Abtastrate einen sehr großen Öffnungswinkel mit sehr hoher Ge-

nauigkeit liefert. Bei einer Abtastrate von 5 Mhz wurde ein durchschnittlicher Fehler von etwa 3 cm bestimmt, was eine genügende Näherung zur Identifikation eines Sprechers in einer Personengruppe liefert. Durch die zugrunde liegende Berechnung über die Kreuzkorrelation lässt sich der Algorithmus sehr gut auf einen FPGA implementieren und parallelisieren. Bedingt durch das Pipeline-Verhalten des FPGAs können die Kreuzkorrelationen parallel für alle Mikrophonpaare ausgeführt werden und haben eine konstante Laufzeit. Die Berechnung des dreidimensionalen Punktes der Schallquelle ist ohne hohen Berechnungsaufwand möglich. Eine Abtastrate von 5 Mhz stellt für ein eingebettetes System wie einen FPGA oder Zynq keine große Herausforderung dar. Damit ist dieser Algorithmus prädestiniert für die Anwendung auf mobilen Robotern. Im nächsten Schritt muss der Algorithmus unter realen Bedingungen getestet und evaluiert werden. Hierfür ist eine Realisierung auf einem Zedboard sowie damit verbundene Messungen im Labor- und Museumsumfeld geplant.

Literatur

1. Bennewitz, M., et al.: Fritz-a humanoid communication robot. In: Robot and Human interactive Communication, 2007. RO-MAN 2007. The 16th IEEE International Symposium on. pp. 1072–1077. IEEE (2007)
2. Faber, F., Bennewitz, M., et al.: The humanoid museum tour guide robotinho. In: Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on. pp. 891–896. IEEE (2009)
3. Marković, I., et al.: Speaker localization and tracking with a microphone array on a mobile robot using von Mises distribution and particle filtering. Robotics and Autonomous Systems 58(11), 1185–1196 (2010), http://dx.doi.org/10.1016/j.robot.2010.08.001
4. Saruwatari, H., et al.: Blind source separation based on a fast-convergence algorithm combining ICA and beamforming. IEEE Transactions on Audio, Speech and Language Processing 14(2), 666–678 (2006)
5. Shiwa, T., et al.: How quickly should communication robots respond? In: Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on. pp. 153–160. IEEE (2008)
6. Strömbergsson, S., et al.: Timing responses to questions in dialogue. In: INTERSPEECH. pp. 2584–2588 (2013)
7. Thrun, S., et al.: Probabilistic algorithms and the interactive museum tour-guide robot minerva. The International Journal of Robotics Research 19(11), 972–999 (2000)
8. Valin, J.M., et al.: Robust sound source localization using a microphone array on a mobile robot. In: Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on. vol. 2, pp. 1228–1233. IEEE (2003)
9. Wang, Z., et al.: Automatic Multi-Speaker Speech Recognition System Based on Time-Frequency Blind Source Separation under Ubiquitous Environment. In: 9th Conference on Industrial Electronics and Applications (ICIEA). pp. 101–106 (2014)
10. Wildermoth, B., Paliwal, K.K.: Use of voicing and pitch information for speaker recognition. SST 2000. Proceedings of the 8th Australian International Conference on Speech Science and Technology (1), 324–328 (2000)

Reinforcement Learning with Object Localization in a Virtual Environment

Joseph Gussev, Helge Ü. Dinkelbach, Frederik Beuth, and Fred H. Hamker

Technische Universität Chemnitz, Artificial Intelligence,
Strasse der Nationen 62, 09111 Chemnitz, Germany
jgus@hrz.tu-chemnitz.de

Abstract. Usually in reinforcement learning scenarios with complex environments, the agent is dependent on detailed external information, for example the location of objects to interact with. An idea would be to let the agent extract needed information from stimuli by itself. This study focuses on reinforcement learning in a virtual environment. It uses $Q(\lambda)$-learning with the main goal being to let an agent learn a goal-directed behavior. The agent learns to find relevant objects in its environment to be able to interact with them, using an attention-driven object localization module. Latter uses a recently developed model of the visual cortex and simulates the attentional processing of the primate brain. The agent in the implemented system is successfully able to use the ability of object localization in a virtual environment.

Keywords: reinforcement learning, virtual environment, object localization, visual attention

1 Introduction

Object recognition and reinforcement learning belong to the broad field of artificial intelligence, yet they are very different. While object recognition is a sub-domain of computer vision, reinforcement learning is a sub-domain of machine learning. Reinforcement learning methods are well discussed in different fields of application, for example robotics [1, 2], industrial manufacturing [3, 4] and computer game playing [5]. Many of these algorithms have a restricted sense or understanding of their environment, for example visual information about objects close to the agent. This work aims to provide the agent with the ability of object recognition, more specifically object localization, to close this gap. Instead of being dependent on detailed external information about the object, the agent would be able to extract needed information by itself from an image provided by a virtual environment. The goal of this study is to let an agent learn a goal-directed behavior, which is to interact with certain objects in a virtual environment. The system to fulfill this task consists of three essential components: reinforcement learning, object localization and a virtual environment. The attention-driven object localization system uses a recently developed model of the visual cortex and simulates the attentional processing of the primate brain.

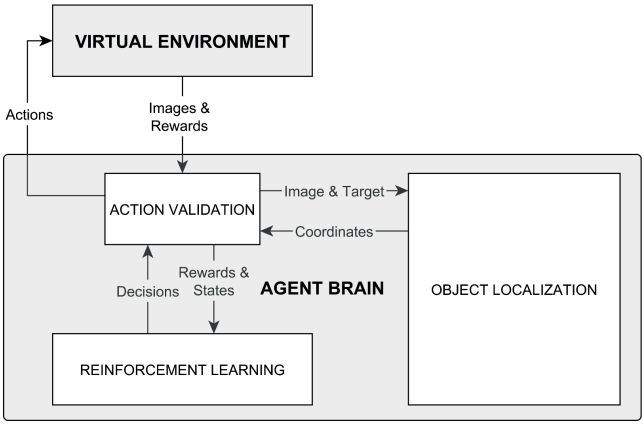


Fig. 1. An overview on the system structure and the information exchange between its components.

2 System Overview

The structure of this study's system as well as the information exchange between its components, which will be explained more detailed in this section, are illustrated in Fig. 1.

Reinforcement Learning is a class of iterative learning algorithms maximizing a long-term reward. It has been shown, for discrete state and action spaces, these algorithms can find optimal action policies for a given problem [6]. In most application scenarios, the reward occurs only after the task is completed, denoted by a terminal state. To deal with this delayed reinforcement, the eligibility trace mechanism [7] can be used. Every time an action is chosen, a trace is created for the corresponding action-state pair, which decays over time during the current trial. $Q(\lambda)$-learning [6] combines both Q-learning [8] and eligibility traces. Latter differs between two types: In accumulating traces [6], the trace gets accumulated each time the according action-state pair occurs. On the other hand, replacing traces [9] set the trace to an initial value each time instead of accumulating it. Whenever a trial ends or an exploratory action was chosen, all traces are deleted. One has to decide depending on the state and action space as well as the reward function which mechanism fits the scenario best. This study uses $Q(\lambda)$-learning with accumulating traces and a softmax action selection [6, 10].

Action Validation: To save up on simulation time as well as prevent the agent from performing absurd actions, for example reaching for an object while already grasping another one, an action validation was implemented. It takes the role

of an interface between the agent and its environment. Based on the concept of an internal environment entity [11], this interface intercepts actions manually characterized as invalid and prevents them from being performed in the virtual environment. Instead, it sends a negative reward to the agent.

Object Localization: The task of object localization is to search a given target object in an image [12]. For that, a recently developed attention model [13] is used. The model is based on the primate visual system of the brain [14, 15], which consists of a bottom-up and top-down processing network. The relevant areas are the prefrontal cortex (PFC), the frontal eye fields (FEF) and the ventral steam, consisting of the primary visual cortex (V1), the fourth visual cortex (V4) and the inferior temporal cortex (IT). Each area encodes different information regarding the object as well as the task. The core of this model is the concept of visual attention, which is a selection process to select relevant information among a large amount of incoming sensory data [16]. Beuth and Hamker [13] propose that attention is a cognitive and holistic top-down control process, tuning the visual system for the current task. What attention does in this model is to amplify the response of neurons encoding task relevant stimuli and to suppress the response of neurons encoding the irrelevant stimuli, therefore adjusting neuronal activity for a current task.

Virtual Environment: To provide the visual stimuli needed for the object localization, the cross-platform game engine Unity was used as a virtual environment. For this study a Unity scene of the EU-project *Spatial Cognition* of the Professorship Artificial Intelligence was used, together with its agent 'Felice'. Using camera objects as eyes, Felice is able to receive visual information in the form of images from the environment. Information exchange between Unity and the rest of the system was done via network.

3 Task

Scenario, States & Actions: Spawning besides an open box the agent will face a table with three different objects on it (Fig. 2). The agent has to put any two objects in the box, one by one, to end a trial. The states and actions were modeled to represent elemental steps of this task. Hence, the agent is able to choose from the following action space:

1) MOVE to table	5) VISUAL SEARCH (1st object)
2) MOVE to box	6) VISUAL SEARCH (2nd object)
3) GRAB center of view	7) VISUAL SEARCH (3rd object)
4) LET go	

A visual search lets the agent fixate its eyes on an object in the image received from the virtual environment. When standing away from the table, the action also sets it as the target. Although fixation is lost when the agent moves towards

Fig. 2. Agent 'Felice' and the three objects chosen for the study. This constellation represents the initial state.

the table, it makes sure the target object is still within the visual field to be fixated again and grabbed. A state is represented by a set of variables:

1) POSITION (1 = at the table; 2 = at the box)
2) HANDS (0 = emtpy; 1-3 = grabbing 1st / 2nd / 3rd object)
3) TARGET OBJECT (0 = emtpy; 1-3 = 1st / 2nd / 3rd object)
4) FIXATED (0 = not fixated; 1 = fixating target object)
5) BOX (0 = emtpy; 1-3 = 1st / 2nd / 3rd object; 4 = two objects)

Reward Function: Whenever the action validation intercepts an invalid action, a reward of $r = -11$ is sent to the agent. The rewards of actions performed within the virtual environment are received from latter itself, only yielding positive reward whenever objects are dropped into the box. Object 1 yields $r = 20$, Object 2 $r = 10$ and Object 3 $r = 30$. As soon as two objects are within the box, signalling the agent's terminal state, the agent receives a final reward consisting of the sum of the rewards of the corresponding objects. Every other action is rewarded with $r = -1$.

Parameters: The $Q(\lambda)$-learning parameters for this scenario were set with learning rate $\alpha = 0.1$, discount $\gamma = 0.7$ and trace decay $\lambda = 0.8$, the softmax parameters with temperature $T = 30$ and exploration rate $\epsilon = 0.2$.

4 Results

Learning converged after about 20 trials: Fig. 3 shows the number of actions as well as the number of invalid actions taken by the agent, per trial. Performing around 150 actions per trial at the beginning, with around 90 of them being invalid, the agent quickly sorts those invalid actions out. The reward of $r = -11$ of invalid actions compared to $r = -1$ of the other actions is responsible for the decrease of invalid actions, since their Q-Values decrease more rapidly with a

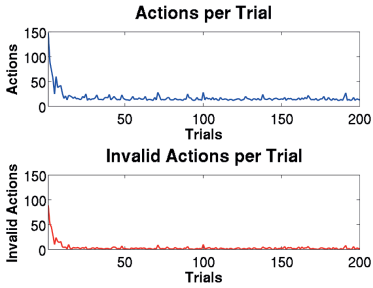


Fig. 3. Actions and invalid actions per trial after 200 trials.

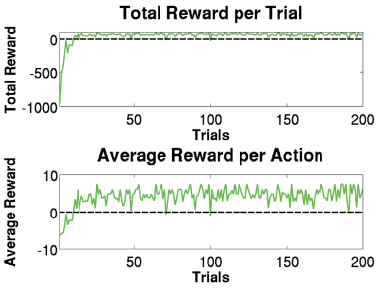


Fig. 4. Total reward and average reward per action after 200 trials.

higher negative reward. After 20 trials, the number of actions per trial averages 25, with only about 0 to 10 being invalid. Latter are still existent because of two reasons: The first is simply exploration. The second one would be, that the states near to the initial state are visited often, so the according Q-Values can be frequently updated for the agent to effectively avoid invalid actions in those states. Meanwhile, the states near the terminal state are visited less, resulting in the Q-Values of the negative actions being not as established. The convergence of learning is also indicated by the total reward and average reward per action, plotted in Fig. 4. Starting with a reward of around -900 per trial and an average reward of -6 per action as a result of the majority of chosen actions being invalid, the agent adjusts to positive rewards in both total reward tending towards the maximum of 90 per trial and average reward per action being at about 6. Since the agent learned to avoid the invalid actions, the total reward per trial as well as average reward per action allow the positive rewards to stand out. This is a clear sign, that the agent learned to go for the profitable actions instead of the invalid and punishing ones. The behavior the agent learned and will use when choosing greedily after 200 trials is represented by the following action chain: 7, 1, 7, 3, 2, 4, 5, 1, 5, 3, 2 and 4. This results in the maximum total reward and average reward per action and shows, that the agent learned an optimal behavior.

5 Conclusion & Outlook

Concluding this work, the idea of a system combining reinforcement learning, a virtual environment and object localization was implemented. The Q-Values converged towards the intended behavior using $Q(\lambda)$-learning with accumulating traces and a softmax action selection mechanism. A possible idea for future research based on this work would be replacing object localization with object detection and accordingly creating a more complex state and action space with an active change of the agent's visual field being possible.

References

1. Takahashi, Y., Asada, M. and Hosoda, K.: Reasonable performance in less learning time by real robot based on incremental state space segmentation. In: Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on, Volume 3, pp. 1518-1524. IEEE. (1996)
2. Takahashi, Y. and Asada, M.: Vision-guided behavior acquisition of a mobile robot by multi-layered reinforcement learning. In: Intelligent Robots and Systems, 2000. (IROS 2000). Proceedings. 2000 IEEE/RSJ International Conference on, Volume 1, pp. 395-402. IEEE. (2000)
3. Ueda, K., Hatono, I., Fujii, N. and Vaario, J.: Reinforcement Learning Approaches to Biological Manufacturing Systems. CIRP Annals Manufacturing Technology, Volume 49, pp. 343-346. (2000)
4. Mahadevan, S. and Theocharous, G.: Optimizing Production Manufacturing using Reinforcement Learning. In: Proc. 11th International FLAIRS Conference, pp. 372-377. AAAI Press. (1998)
5. Defazio, A. and Graepel, T.: A Comparison of learning algorithms on the Arcade Learning Environment. arXiv preprint arXiv:1410.8620 (2014)
6. Sutton, R. S. and Barto, A. G.: Reinforcement Learning: An Introduction. Cambridge, MA: MIT Press. (1998)
7. Klopf, A. H.: Brain function and adaptive systems: a heterostatic theory. Technical Report AFCRL-72-0164. Air Force Cambridge Research Laboratories. (1972)
8. Watkins, C. J. C. H.: Learning from delayed rewards. PhD Thesis, University of Cambridge, England (1989)
9. Singh, S. P. and Sutton, R. S.: Reinforcement Learning with Replacing Eligibility Traces. Machine Learning, Volume 22, pp. 123-158. Kluwer Academic Publishers-Plenum Publishers. (1996)
10. Tan, M.: Multi-Agent Reinforcement Learning: Independent vs. Cooperative Agents. In: Proc. 10th International Conference on Machine Learning, pp. 330-337. (1993)
11. Singh, S. P., Barto, A. G. and Chentanez, N.: Intrinsically motivated reinforcement learning. Saul, L. K., Weiss, H. and Bottou, L. (Eds.), Advances in Neural Information Processing Systems, Volume 17, pp. 1281-1288. MIT Press. (2005)
12. Wolfe, J. M.: Guided search 2.0 a revised model of visual search. Psychonomic Bulletin & Review, Volume 1, pp. 202-238. (1994)
13. Beuth, F. and Hamker, F. H.: Attention as cognitive, holistic control of the visual system. In: Villmann, T. and Schleif, F. M. (Eds.), Proc Workshop New Challenges in Neural Computation 2015 - NCNC 2015, Machine Learning Reports 03/2015,pp. 133-140. (2015)
14. Jamalian, A. and Hamker, F. H.: Biologically Inspired Models for Attentive Robot Vision: A Review. In: Pal, R. (Ed.), Innovative Research in Attention Modeling and Computer Vision Applications, pp. 69-98. Information Science Reference. (2016)
15. Serre, T.: Learning a dictionary of shape-components in visual cortex: Comparison with neurons, humans and machines. PhD Thesis. (2006)
16. Carrasco, M.: Visual attention: the past 25 years. Vision Research, Volume 51, pp. 1484-1525. (2011)

Automatische Berechnung des Ruiz-Kriteriums im Finite-Elemente-Postprocessing

Hannes Kuhnert

hannes.kuhnert@s2003.tu-chemnitz.de

Zusammenfassung: Reibdauerermüdung ist einer der wesentlichen im Maschinenbau bekannten Schädigungsmechanismen. Der Fretting Fatigue Damage Parameter, auch als Ruiz-Kriterium bekannt, bietet in der Simulation von Reibdauerermüdungsvorgängen eine Möglichkeit zur Einschätzung der lokalen Anrisswahrscheinlichkeiten. Der vorliegende Artikel skizziert die Entwicklung einer Ergebnisaufbereitungsroutine für das Finite-Element-Programm Abaqus, welche für beliebige Kontaktsimulationen das Ruiz-Kriterium nach Ziaei berechnet. Es werden Fragen der mathematischen Umsetzung, des Datenzugriffs und der Integration in die grafische Oberfläche von Abaqus/CAE behandelt. Abschließend werden beispielhaft zwei Anwendungen vorgestellt.

Schlagwörter: Tribologie, Reibdauerermüdung, Fretting, Ruiz-Kriterium, FFDP, Finite-Elemente-Methode, FEM, Postprocessing, Abaqus, Python

1 Reibdauerermüdung

Reibdauerermüdung ist im Maschinenbau ein bedeutendes Phänomen und bei vielen Erzeugnissen für Betriebssicherheit und Lebensdauer entscheidend. Reibdauerermüdung ist Verschleiß unter schwingender Oberflächenbeanspruchung. Dazu gehört die Bildung von Mikroverschweißungen und ihre Abscherung, plastische Verformung, Abrasion, Oxidation und die Bildung von Mikro- und Makrorissen. Insbesondere die Entstehung eines Makrorisses bedeutet gewöhnlicherweise das Versagen des Bauteils. [PM78] [GF05] Abbildung 1 veranschaulicht das Prinzip des Reibkontakts und zeigt einige hierbei wichtige mechanische Größen.

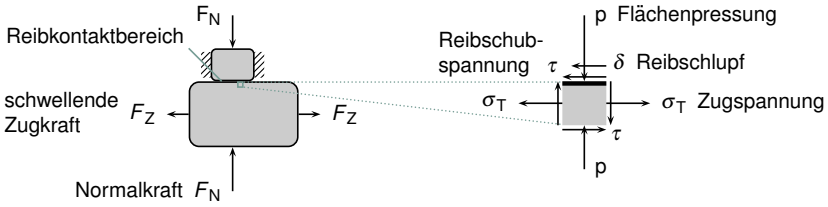


Abbildung 1: generischer Reib-Sachverhalt, rechts: differenzieller Ausschnitt eines Kontaktpartners mit Schnitt- und Reaktionsgrößen

2 Das Ruiz-Kriterium

Zur Berücksichtigung der Reibdauerermüdung in der Konstruktion stehen aufgrund der Vielfalt und Komplexität der Prozesse keine universellen, das Problem abschließend behandelnden Berechnungsansätze zur Verfügung. Denoch lassen sich gewisse allgemeine Prinzipien zur rechnerischen Analyse ausnutzen, bspw. der Einfluss der eingebrachten Reibarbeit. Ein solcher Ansatz wurde in den 1980er-Jahren von Ruiz et al. mit dem *Fretting Fatigue Damage Parameter* (FFDP), auch als Ruiz-Kriterium bekannt, vorgestellt. [RBC84] Dabei handelt es sich um einen halbempirischen qualitativen Kennwert. Er gilt als Indikator für die lokale Anrisswahrscheinlichkeit, gibt also einen Hinweis auf die wahrscheinlichsten Anrissorte. Die originale Formulierung für das Ruiz-Kriterium k lautet:

$$k = \sigma_{\mathrm{T}} \cdot \tau \cdot \delta \tag{1}$$

Hierbei ist $\tau \cdot \delta$ die in die Oberfläche eingebrachte (flächen-)spezifische Reibarbeit, gebildet als Produkt aus Reibschubspannung τ und Reibschlupf δ. Die eingetragene spezifische Reibenergie korreliert mit der Reibdauerermüdung der Oberfläche. Die tangentiale Zugspannung σ_{T} kann in einer derart vorgeschädigten Oberfläche dazu führen, dass aus Mikrorissen Makrorisse entstehen.

Gleichung 1 geht von einer zweidimensionalen Auffassung des Reibvorgangs mit linienförmig betrachtetem Kontaktbereich und einer Kontaktfreiheit aus. Ziaei hat das Ruiz-Kriterium in Richtung einer breiteren Nutzbarkeit weiterentwickelt. [Zia02] Dies beinhaltet eine veränderte Formulierung der spezifischen Reibarbeit und eine abweichende Zugspannungsbehandlung. Die Reibarbeit wird differenziell betrachtet und als Skalarprodukt vektorieller Größen formuliert. Statt der tangentialen Zugspannung wird die erste Hauptspannung verwendet. Das modifizierte Ruiz-Kriterium (mFFDP) ist für beliebige dreidimensionale Reibermüdungs-Sachverhalte nutzbar. Im Folgenden bezieht sich der Begriff *Ruiz-Kriterium* stets auf diese nach Ziaei modifizierte Formulierung:

$$mFFDP = w_{\mathrm{R}} \cdot \sigma_1 \tag{2}$$

$$= \int \vec{\tau} \cdot \mathrm{d}\vec{\delta} \cdot \sigma_1 \tag{3}$$

Da das Ruiz-Kriterium auf lokalen Größen in der Kontaktzone beruht, die in der Realität nicht gemessen werden können, bilden Simulationen nach der Finite-Elemente-Methode (FEM) die Grundlage seiner Anwendung. Dabei reicht, weil das Ruiz-Kriterium eine rein qualitative Größe ist, die Simulation eines einzelnen Lastwechsels aus. Von einem im Betrachtungshorizont unverändert ablaufenden Reibvorgang ausgehend charakterisiert der spezifische Eintrag von Reibarbeit w_{R} an der Oberfläche nach einem Lastwechsel gleichfalls die Verteilung der eingebrachten Reibarbeit nach beliebig vielen Lastwechseln. Was die Zugspannung betrifft, ist für jeden Ort der größte im Lastwechsel auftretenden Werte bedeutsam. Eine zweite potientiell aussagekräftige Variante ergibt sich mit der Differenz von größtem und kleinstem Wert.

$$mFFDP_{\mathrm{max}} = w_{\mathrm{R}} \cdot \sigma_{1\,\mathrm{max}} \tag{4}$$

$$mFFDP_{\mathrm{amp}} = w_{\mathrm{R}} \cdot (\sigma_{1\,\mathrm{max}} - \sigma_{1\,\mathrm{min}}) \tag{5}$$

3 Automatische Berechnung des Ruiz-Kriteriums

Am Lehrstuhl für Konstruktionslehre ist bei bisherigen Anwendungen in der Forschung das Ruiz-Kriterium manuell berechnet worden. Nachdem jeweils mit dem Finite-Elemente-System Abaqus eine Simulation durchgeführt worden war, wurden den Simulationsergebnissen händisch die Ausgangsgrößen für das Ruiz-Kriterium entnommen. Diese wurden dann mit Hilfe einer komplexen Kalkulationstabelle verrechnet, um das Ruiz-Kriterium zu erhalten. Dieses sehr aufwendige Verfahren ist für dreidimensionale Modelle und Modelle mit komplizierter Geometrie praktisch nicht brauchbar. Weiter ist festzustellen, dass das Ergebnis, da es von der Modellgeometrie losgelöst tabellarisch errechnet wird, nicht auf der Modellgeometrie visualisiert werden kann.

Daher lag es nahe, eine automatischen Finite-Elemente-Postprocessing-Routine für das Ruiz-Kriterium zu entwickeln. Ziel war also eine Routine, die für das FE-Modell eines Kontaktproblems nach einer durchgeführten Simulationsrechnung die Ergebnisse aufbereitet, um das Ruiz-Kriterium zu erhalten. Dies sollte die Ermittlung desselben erheblich erleichtern und zuverlässiger machen, seine Nutzung für komplexere und – sofern technisch umsetzbar – auch für dreidimensionale Modelle ermöglichen und die Ergebnisse idealerweise auf die Modellgeometrie bezogen in der Ergebnisdatei der FE-Simulation ablegen. Die genauen Möglichkeiten waren im Rahmen der Konzeption und Entwicklung zu prüfen.

4 Mathematische Umsetzung

Bei der rechentechnischen Umsetzung des Ruiz-Kriteriums ist die Berechnung der Reibarbeit eine spezielle Betrachtung wert. Die gewählte Lösung ist in [Kuh16] beschrieben und soll hier in strafferer Form dargestellt sein.

Gemäß Gleichung 3 ist die spezifische Reibarbeit als Integral des Skalarprodukts von Reibschubspannung und differenziellem Schlupfweg zu verstehen:

$$w_\mathrm{R} = \int \vec{\tau} \cdot \mathrm{d}\vec{\delta} \tag{6}$$

Die Ergebnisse einer Finite-Elemente-Analyse liegen örtlich und zeitlich diskretisiert vor. Werte für die Reibschubspannungen und für die Schlupfwege existieren nur an den Elementknoten und nur für einzelne Zeitpunkte t_i. Zwischen benachbarten Zeitpunkten t_{i-1} und t_i wird jeweils ein linearer Verlauf angenommen.

Um die diskrete Integration von einem Zeitpunkt zum nächsten als Multiplikation formulieren zu können, sei der Integrationsersatzwert $\bar{\tau}$ eingeführt, zunächst für eine Dimension:

$$\int_{t=t_{i-1}}^{t_i} \tau(t) \cdot \mathrm{d}\delta(t) = \bar{\tau}(t_{i-1}, t_i) \cdot \Big(\delta(t_i) - \delta(t_{i-1})\Big) \tag{7}$$

$\bar{\tau}(t_{i-1},t_i)$ soll also der Integration eines linearen Reibschubspannungsverlaufs mit den Randwerten $\tau(t_{i-1})$ und $\tau(t_i)$ über die Länge 1 entsprechen. Das bedeutet in grafischer Darstellung für die originale Größe τ und die Ersatzgröße $\bar{\tau}$ die Gleichheit der Flächeninhalte zwischen Graph und Abszisse, wie in Abbildung 2 veranschaulicht.

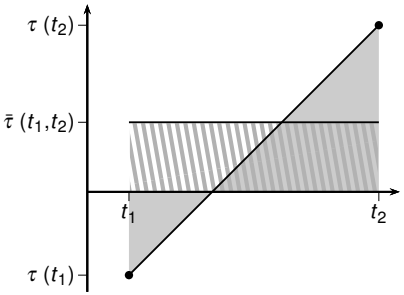


Abbildung 2: Integration der Reibschubspannung τ

Mit dem Ansatz der Gleichheit der Flächeninhalte ergibt sich für $\bar{\tau}$:

$$\bar{\tau}(t_{i-1},t_i) = \frac{\left|\tau(t_{i-1})\right|\left(-\frac{\tau(t_{i-1})}{\tau(t_i)-\tau(t_{i-1})}\right)+\left|\tau(t_i)\right|\left(1+\frac{\tau(t_{i-1})}{\tau(t_i)-\tau(t_{i-1})}\right)}{2} \tag{8}$$

Diese Formulierung des Integrationsersatzwerts $\bar{\tau}$ erübrigt eine explizite Berechnung der Nullstellen von τ, wie sie ansonsten zur korrekten Integration durchgeführt werden müsste.

Mit $\bar{\tau}$ kann das Integral von Gleichung 6 in eine Summe von Produkten finiter Werte überführt werden:

$$\begin{aligned} w_{\mathrm{R}} &= \sum_{i=1}^{n} \vec{\bar{\tau}}(t_{i-1},t_i)\cdot\Delta\vec{\delta}(t_{i-1},t_i) \\ &= \sum_{i=1}^{n} \vec{\bar{\tau}}(t_{i-1},t_i)\cdot\left(\vec{\delta}(t_i)-\vec{\delta}(t_{i-1})\right) \\ &= \sum_{i=1}^{n}\sum_{r} \bar{\tau}_r(t_{i-1},t_i)\cdot\left(\delta_r(t_i)-\delta_r(t_{i-1})\right) \end{aligned} \tag{9}$$

Dies gilt für jeden Knoten des FE-Modells. Mit dem Index i wird über die Zeitinkrement des Modells summiert, n ist die Zahl der Inkremente. Der Index r steht für die Richtungen der Freiheiten in der Kontaktfläche. Bei dreidimensional modellierter Geometrie kann er die Werte 1 und 2 annehmen, bei zweidimensionaler Modellierung gibt es nur eine Richtung, mit der Benennung 1.

5 Technische Realisierung

Der Algorithmus der Berechnung des Ruiz-Kriteriums durch Umsetzung der Gleichungen 9, 8, 4 und 5 ist in Abbildung 3 veranschaulicht.

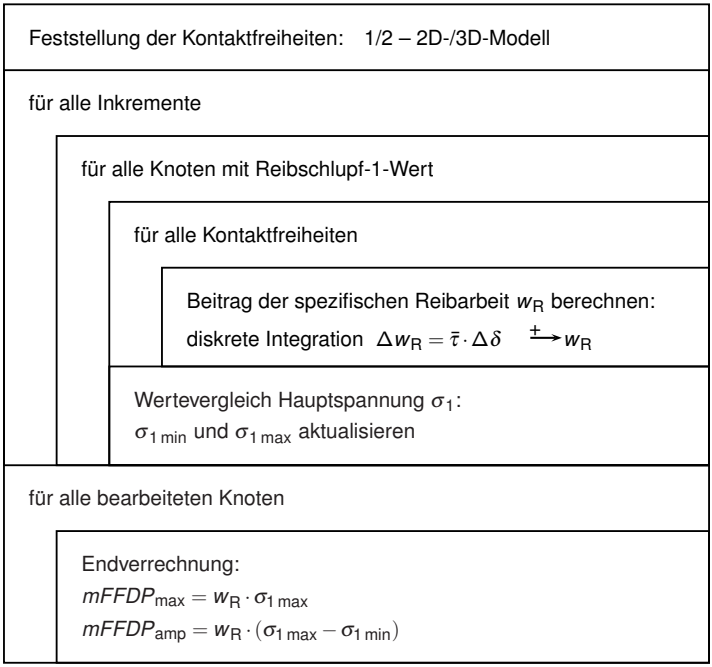


Abbildung 3: prinzipielle Darstellung des Ablaufs der Berechnung des Ruiz-Kriteriums

Für die technische Umsetzung impliziert das Ziel, das Ruiz-Kriterium mit der Modellgeometrie verknüpft vorliegen zu haben, eine Bindung an das Erweiterungs-Interface des verwendeten Finite-Elemente-Systems. Abaqus bietet ein solches Interface auf Basis der Skriptsprache Python an. Die Struktur der entstandenen Berechnungsroutine ist in hohem Maße von den Möglichkeiten dieses Interfaces geprägt.

Abaqus ist auf der Kommandozeile verfügbar, bietet aber auch, mit Abaqus/CAE, eine grafische Oberfläche. Die eigentlichen Daten sind bei beiden Verwendungsweisen gleichartig über ein Objekt-Modell les- und manipulierbar. Bei der Entwicklung der Benutzer-Interaktion erfolgte, da seitens der Auftraggeber vorwiegend mit Abaqus/CAE gearbeitet wird, zunächst eine Beschränkung auf die grafische Oberfläche von Abaqus/CAE.

Abaqus/CAE besteht zur Laufzeit aus zwei separaten Prozessen, dem Kern und der grafischen Oberfläche (GUI). Während der Kern die Modell- und Ergebnisdaten verwaltet und auch für die grafische Anzeige derselben verantwortlich ist, bildet der GUI-Prozess die eigentliche grafische Nutzeroberfläche. Das GUI bietet über einen Plug-in-Mechanismus

eng abgegrenzte Möglichkeiten zur Erweiterung der Funktionalität. Dieser Mechanismus wurde hier genutzt, um im Hauptmenü von Abaqus/CAE die Funktion der Berechnung des Ruiz-Kriteriums zur Verfügung zu stellen. [Das14d] [Das14b]

Die Struktur dieses Plug-ins innerhalb der Abaqus/CAE-Umgebung ist in Abbildung 4 veranschaulicht. Während zur Interaktion mit dem Nutzer im GUI programmiert werden muss, liegen die eigentlich zu bearbeitenden Daten im Kern-Prozess vor. Zwar sind vom GUI aus über das Modul *kernelAccess* Zugriffe auf die Daten des Kerns möglich. Diese Methode ist jedoch wegen ihrer Aufwendigkeit nicht zur Durchführung umfangreicher Berechnungen geeignet. Daher erfolgt durch das GUI-Plug-in nur die generelle Ablaufsteuerung, die Vorprüfung des Datenbestands und die Interaktion mit dem Benutzer, während für die eigentliche Berechnung eine separate Routine aufgerufen wird, die selbstständig im Kern läuft.

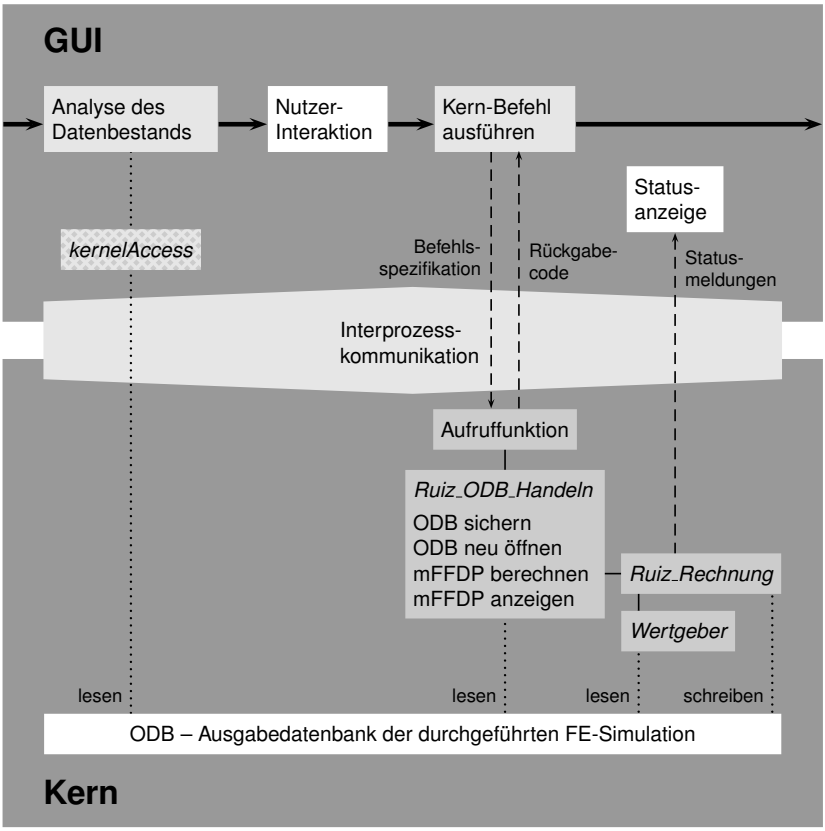


Abbildung 4: Strukturschema des Abaqus/CAE-Plug-ins für das Ruiz-Kriterium

In der Interaktion mit dem Nutzer ist vor allem die Wahl des zu bearbeitenden Lastschritts von Bedeutung. Ein FE-Modell kann neben einem Lastschritt, der für eine Dauerbelastungs steht, vorbereitende Lastschritte enthalten. Ebenfalls möglich ist, dass ein FE-Modell eine Mehrzahl von Lastschritten, die jeweils für einen Dauerbelastungsvorgang stehen, enthält. Daher muss vom Nutzer spezifiziert werden, welcher Lastschritt zu bearbeiten ist. Abbildung 5 zeigt das Dialogfenster mit der Lastschrittauswahl und weiteren Optionen. Wird ein Lastschritt gewählt, für den das Ruiz-Kriterium weder berechnet werden kann, noch bereits berechnet ist, wird die Eingabe nicht angenommen. Bei Wahl eines Lastschritts, für das die Berechnung schon durchgeführt worden ist, wird das vorhandene Ergebnis zur Anzeige gebracht. Damit ist es unabhängig davon, ob das Ruiz-Kriterium bereits berechnet worden ist, auf gleiche Weise möglich, über den Aufruf der Berechnungsfunktion Zugang zum Ruiz-Kriterium zu erhalten.

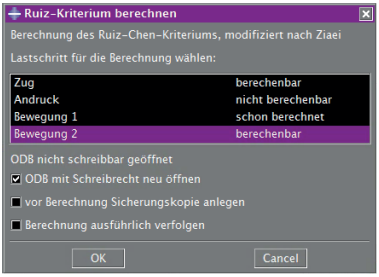


Abbildung 5: Nutzerdialog – Auflistung der Lastschritte des Modells

Abbildung 4 zeigt neben dem vom GUI-Prozess von Abaqus/CAE kontrollierten Gesamtablauf der Plug-in-Ausführung auch die Struktur des im Kern laufenden Teils des Plug-ins.

Die Plug-in-Architektur gibt hier den Aufruf einer Funktion mit in Textform übergebbaren Parametern vor. Für die weitere Struktur wurde eine objektorientierte Lösung gewählt. Die Aufruffunktion erzeugt zunächst ein Objekt der Klasse *Ruiz_ODB_Handeln* und ruft an diesem entsprechend der empfangenen Parameter die Funktionen für das Sichern oder Neuöffnen der zu bearbeitenden Ausgabedatenbank sowie das Berechnen und/oder Anzeigen des Ruiz-Kriteriums auf.

Die Berechnung des Ruiz-Kriteriums ist – nach dem in Abbildung 3 dargestellten Algorithmus – in der Klasse *Ruiz_Rechnung* im Zusammenwirken mehrerer Klassenfunktionen implementiert.

Dem Zugriff auf die Ausgangsdaten der Berechnung dient die Klasse *Wertgeber*. Im Hintergrund steht der Umgang von Abaqus mit einem für die Finite-Element-Methode spezifischen Datenstrukturproblem: Die Berechnung des Ruiz-Kriteriums erfolgt knotenweise. Die Kontaktgrößen Reibschubspannung τ und Reibschlupf δ sind in der Finite-Elemente-Analyse Knotengrößen. Dagegen liegen die allgemeinen Spannungen im Modellgebiet originär nicht an den Knoten, sondern an den Integrationspunkten der Elemente vor. Werden Hauptspannungen für Knoten abgefragt, werden diese Werte durch Extrapolation über die Elementansatzfunktionen gewonnen. Es hat sich gezeigt, dass Abaqus bei jeder Abfra-

ge eines Knotenwerts solch einer Integrationspunktgröße (erneut) eine Extrapolation des gesamten Knotenfelds der Größe vornimmt. Dies bringt einen hohen Rechenzeitbedarf mit sich. Überdies wurde ein erhebliches Anwachsen der bearbeiteten Ausgabedatenbank beobachtet, ohne dass ein zusätzlicher Datenbestand sichtbar wäre. Dies ist ein Indiz dafür, dass alle extrapolierten Knotenwertfelder in der Ausgabedatenbank aufgebaut werden und dort referenzlos verbleiben. Die Abaqus-Dokumentation benennt die prinzipiellen Datenzugriffsmöglichkeiten, stellt hierzu aber keine Details dar. [Das14c] Mit Hilfe der Klasse *Wertgeber* wird das beschriebene Problem umgangen. In der Klasse *Ruiz_Rechnung* werden alle Anfragen für Knotenwerte an ein *Wertgeber*-Objekt gerichtet. Bei Größen, die originär an den Integrationspunkten vorliegen, speichert das *Wertgeber*-Objekt ein extrapoliertes Feld für alle Knoten des Modells zwischen. Werden anschließend Anfragen für andere Knotenwerte der gleichen Größe im gleichen Inkrement empfangen, werden die Werte dem zwischengespeicherten Feld entnommen. Mit diesem Mechanismus wird die mehrfache Extrapolation von Knotenwertfeldern von Integrationspunktgrößen vermieden.

6 Anwendungsbeispiele

Eine erste Anwendung fand mit Versuchen auf dem Reibklötzchenprüfstand der Professur Konstruktionslehre statt. Auf diesem Prüfstand können grundlegende Reibdauerermüdungs-Untersuchungen mit verschiedenen Materialpaarungen und gezielter Variation der mechanischen Parameter durchgeführt werden. [VL12] Im Versuch werden an eine Flachprobe beidseitig Reibklötzchen angedrückt, die Flachprobe wird schwellend zugbelastet und zusätzlich kann auf die Reibklötzchen eine Schwingbewegung aufgebracht werden. Durch eine Kraft- und Wegüberwachung und -regelung können damit die Zugbelastung der Probe und der Reibschlupf frei variiert werden. Das Versuchsschema ist in Abbildung 6 links dargestellt, in seitlicher Ansicht hälftig mit Symmetrieachse. Für Reibversuche auf diesem Prüfstand ist zu erwarten, dass das Ruiz-Kriterium Aussagen über den Anrissort treffen kann. Abbildung 6 zeigt fünf Proben nach Reibklötzchenversuchen. Abbildung 7 stellt für einen Reibklötzchenversuch das modifizierte Ruiz-Kriterium auf Grundlage einer zweidimensionalen FE-Analyse dar. Die im Diagramm verwendete x-Achse läuft längs der Flachzugprobe und ist in Abbildung 6 wiederzufinden.

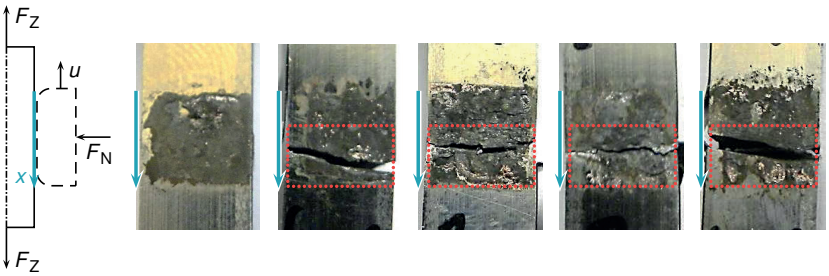


Abbildung 6: Proben aus Reibklötzchenversuch; ganz links Skizze des Versuchsprinzips, 90° um die Längsachse gedreht

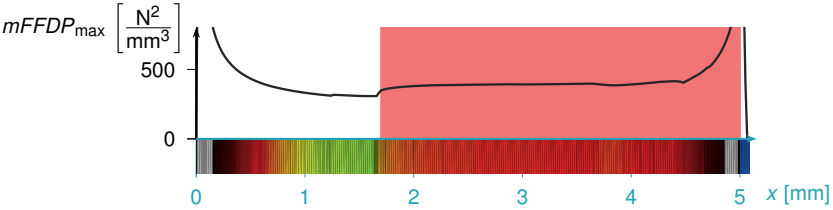


Abbildung 7: Reibklötzchenversuch: modifiziertes Ruiz-Kriterium nach Ziaei im Diagramm, vgl. x-Achse in Abbildung 6

Das Ruiz-Kriterium $mFFDP_{max}$ zeigt an den Rändern extreme Ausschläge. Diese sind auf numerische Singularitäten in der FE-Analyse zurückzuführen und physikalisch nicht bedeutsam. Relevante erhöhte Werte zeigen sich jedoch für x-Werte über 1,7 mm. Der Versuchsaufbau ist im Grundansatz auch in Längsrichtung symmetrisch, was zunächst einen ebensolchen Verlauf des Ruiz-Kriteriums erwarten lässt. Jedoch weist die Konstruktion der Reibklötzchen-Einspannung – hier nicht abgebildet – eine leichte Asymmetrie in Einspannkräften und Steifigkeit auf. Das verwendete FE-Modell bildet dies ab und es ergeben sich entsprechende Werte auch für das Ruiz-Kriterium. Der Bereich erhöhter Werte ist im Diagramm rot hervorgehoben. Der entsprechende Bereich ist auch in Abbildung 6 an den gerissenen Proben markiert. Es zeigt sich, dass bei allen gebrochenen Proben der Riss im Bereich des erhöhten Ruiz-Kriteriums aufgetreten ist.

Ein Beispiel für die Anwendung an einem dreidimensionalen Modell zeigt Abbildung 8.

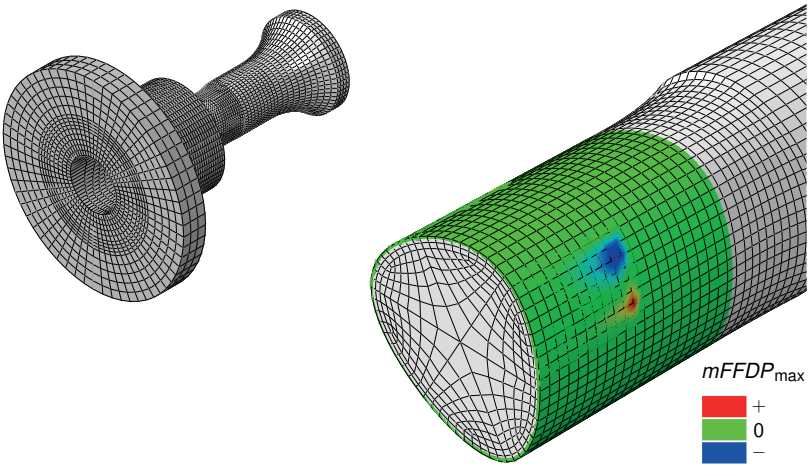


Abbildung 8: FE-Modell einer torsionswechselbelasteten P3C-Welle-Nabe-Verbindung, Welle-Nabe-Modell und freigeschnittene Welle mit visualisiertem Ruiz-Kriterium, FE-Modellierung: F. Reiß, Lehrstuhl Konstruktionslehre

Hierbei handelt es sich um einen Welle-Nabe-Pressverband mit einem P3C-Polygon-Wellenprofil. Bei torsionsbelastete Pressverbände tritt stets Mikroschlupf auf, insofern handelt es sich bei Pressverbänden um reibermüdungsgefährdete Maschinenelemente. Der größte Schlupf ist generell im Bereich der Nabenkante zu erwarten, ebenso die Entstehung von Rissen. Bei unrunden Profilen wie dem vorliegenden Polygonprofil ist darüber hinaus der Ort der Rissinitiierung über dem Umfang von Interesse.

Das FE-Modell weist am dreifach punktsymmetrischen Umfang jeweils an charakteristischen Stellen ein erhöhtes Ruiz-Kriterium auf. Eine sektorielle Betrachtung des Umfangs zeigt eine Übereinstimmung von erhöhtem Ruiz-Kriterium und typischen tatsächlichen Anrissorten in Realversuchen.

7 Fazit

Das Ziel einer automatischen Finite-Element-Postprocessing-Rechenroutine für das Ruiz-Kriterium konnte erreicht werden. Die Routine berechnet an beliebigen zwei- und dreidimensionale Abaqus-Kontaktmodellen das nach Ziaei modifizierte Ruiz-Kriterium. Sie arbeitet zuverlässig und mit hoher Nutzerfreundlichkeit und ist im produktiven Einsatz. Anhand von Beispielen von Reibdauerermüdungsvorgängen wurde die Aussagekraft des Ruiz-Kriteriums für die wahrscheinlichen Anrissorte verdeutlicht.

Literatur

[Das14a] Dassault Systèmes. *Abaqus GUI Toolkit Reference Guide*, 2014.

[Das14b] Dassault Systèmes. *Abaqus GUI Toolkit User's Guide*, 2014.

[Das14c] Dassault Systèmes. *Abaqus Scripting Reference Guide*, 2014.

[Das14d] Dassault Systèmes. *Abaqus Scripting User's Guide*, 2014.

[GF05] K.-H. Grote und J. Feldhusen, Hrsg. *Dubbel – Taschenbuch für den Maschinenbau*. Springer, Berlin Heidelberg New York, 21. Auflage, 2005.

[Kuh16] H. Kuhnert. Verifikation des mFFDP und automatisierte Berechnung für beliebige FE-Kontaktsimulationen in Abaqus, 2016.

[PM78] G. Polzer und F. Meißner. *Grundlagen zu Reibung und Verschleiß*. Deutscher Verlag für Grundstoffindustrie, Leipzig, 1. Auflage, 1978.

[RBC84] C. Ruiz, P. H. B. Boddington und K. C. Chen. An Investigation of Fatigue and Fretting in a Dovetail Joint. *Experimental Mechanics*, 24:208–217, September 1984.

[VL12] J. Vidner und E. Leidich. Untersuchungen zur Auswahl und Modifizierung eines auf dem Spannungsmodell basierenden Reibdauerfestigkeitskritieriums für ausgewählte Maschinenelemente und Ermittlung von Grenzwerten zur Entwicklung eines Reibdauerfestigkeitskonzeptes. Abschlussbericht. DFG-Forschungsvorhaben LE 969/14, 2012.

[Zia02] M. Ziaei. Analytische Untersuchung unrunder Profilfamilien und numerische Optimierung genormter Polygonprofile für Welle-Nabe-Verbindungen, 2002.

Learning of Spatial Invariances for Object-Ground Separation

Alex Schwarz, Frederik Beuth, Fred H Hamker
{alexschw,beuth,fhamker}@hrz.tu-chemnitz.de

TU Chemnitz, Professorship for Artificial Intelligence

Abstract. Whenever a biological plausible model should learn object representations in a real world scenario, the model tends to learn connections not only to features of the learned object but also to some features of the background. We present a learning algorithm that is capable of learning background-independent object representations due to temporal differences in the background. The algorithm is based on a previous one, used for learning the object representations in the visual attention model of Beuth and Hamker [2]. As the representations in this model have encoded a black background among the object, the model performed very well when localizing an object before a black-background, but only averagely when the object is placed in a real-world scene. To address such problems, we advance the learning algorithm to obtain background-independent object representations, and adapt the V1-layer to generate responses similar to their biological counterparts.

1 Introduction

Many biological plausible models of visual attention exist, but only a few have been shown with real-world scenes [1, 3, 6, 11]. It is still unsolved how objects must be represented to allow such models an operation in real world scenarios. Beuth and Hamker(2015) presented an attention model capable of operating on real-world scenes [2]. This model learned the object representations with the principle of temporal continuity [4, 10], as some authors suppose that the primate brain uses this mechanism for the development of invariant objects representations. Temporal continuity learning exploits the fact that, on a short time scale, changes in the visual input are more likely to originate from the same object, rather than from a different one. Temporal continuity learning algorithms have been typically used in simple scenes consisting of black bars, but not in a real-world task. In previous works [1, 2], the learning was already improved to operate in natural scenes by introducing a high post-synaptic threshold. Such a threshold allows to learn cells which are highly specific for their encoded stimuli, which perform well in setups with natural scenes. However, these object representations are still not encoding the object independently of the background. The reason is that the objects are learned from small image patches which contain the object along with some background. As the procedure learns an object representation from all available features in such a patch, it learns the objects

along with the background. However, an object representation independent of the background would be preferable. It is currently unclear how the human brain learns background invariant object representations. Suggestions cover the usage of disparities [5] or motion [9], whereas we show here that temporal continuity alone is powerful enough.

In this article, we will firstly explain the existing model and learning algorithm [2], before outlining its disadvantages and presenting our solution. Secondly, we will show a brief overview of the implemented model mechanisms. Finally, we will show the details of the novel learning algorithm and demonstrate its performance by a miniature version of the model.

2 Model

2.1 The attention model

The model of Beuth and Hamker [2] consists of several layers representing the areas of the visual cortex. The first and lowest layer is the V1-layer, encoding color differences and simple orientations. The latter is constructed out of differently oriented Gabor-Filters, functions of a cosine and a Gaussian. Gabor-Filters are used because their shape fits the shape of typical receptive fields of V1-neurons [7]. Every modeled V1-neuron is then connected to a layer of HVA-neurons. These neurons of a higher visual area (HVA) are a representation of neurons in area V4 and the inferior-temporal cortex IT. In these cells an object representation is constructed out of the responses of the V1-neurons. A feature-based attention signal is then send from the prefrontal cortex (PFC) to the object-view specific cells in HVA, altering the strength of their response. The loop to the frontal eye field (FEF) will then find the correct location of the given object and propagate it back to the corresponding location in HVA.

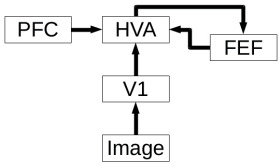


Fig. 1. Schematic layout of the model, showing the main sequence of visual areas working together

2.2 The existing temporal-continuity learning

The construction of feature specific HVA-cells is performed by learning the individual weights from each V1-cell to each HVA-cell with a temporal-continuity learning algorithm (see eq. 1)

$$\tau_w \frac{\partial w}{\partial t} = u_t \cdot v_{t-1} - \alpha \cdot w \cdot v_{t-1}^2 \qquad with: u = \left(r^{\mathrm{V1}} - \theta^{\mathrm{V1}}\right),\ v = \left(r^{\mathrm{HVA}} - \theta^{\mathrm{HVA}}\right)^{+} \tag{1}$$

where r^{V1} and r^{HVA} are the responses of the presynaptic V1-cell and the post-synapic HVA-cell with their respective thresholds θ^{V1} and θ^{HVA}, τ_w being the

learning rate and α the strength of the alpha-constraint. The alpha-constraint is an inhibitory term for the normalization and soft-limitation for the strength of weights (w) between the pre- and postsynaptic cells. The $()^+$ is a symbol for a cutoff of negative values.

2.3 The novel temporal-continuity learning algorithm

When using the temporal-continuity learning algorithm (section 2.2) to achieve background-invariant object representation, the objects are represented in the connections from V1 to HVA. The temporal continuity alone is theoretically powerful enough to learn background invariant HVA-cells on any given background. As the background changes much more often than the object, the learning rule should not learn connections from the background region, and weights only from the features of an object. Yet, we observed that inhibitory weights to the background were learned mistakenly. We found out that the reason is a difference in the learning speed for weights with positive or negative weight changes. Due to this, we introduce two normalization procedures of τ to ensure that the learning speed for both cases is balanced and that weights are generally faster decreasing towards zero (Sec. 4).

3 Implementation Details

3.1 Input Images

We use an image library, consisting of images from 100 objects in 72 different angles (COIL-100, [8]), to learn object-relevant features based on real data. To enable the learning of background-invariant cells, we need to expand the object dataset to show the objects not only on black backgrounds, but also on differently colored backgrounds, gaussian-white noise and patches from real world scenes. We systematically vary the background for each object view, using 1 black, 3 gaussian-white noise, 7 colored, and 9 random real world backgrounds (Fig. 2). We used 15 out of these 100 objects for a faster learning, thereby resulting in 21600 input images ($21600 = (1 + 3 + 7 + 9) \cdot 15 \cdot 72$).

Fig. 2. Examples of the used object patches, showing one black, two colored, a noisy and two real world patches from left to right.

3.2 Normalization of the V1-cell responses

Cells of area V1 are responding equally to different lighting conditions since the retinal receptors habituate to changing intensities of light. Our model should

be able to mimic this behavior. That is accomplished by the V1-cells depending on differences in contrast instead of absolute intensity values. Therefore we normalized the responses of the V1-neurons to roughly fit a range of 0 and 1.

3.3 Filter for equiluminant borders

The gabor filters in the previous model have been run on a gray-scaled version of the image. Human vision on the other hand is still able to extract features of objects and classical borders, even when receiving a colored border which is equiluminant and thus does not appear in gray-scale. To model this ability, we let the model receive input from border-detecting cells for different colors, and combine their input to an equiluminant input. As a result these border cells increased the identification performance, when added to the color based filters.

3.4 A miniature version of the learning algorithm

We included a miniature version of the learning algorithm for demonstrate the effects of several parameters easily. The miniature model consists of all features of the complete model, but does not receive any image as an input. Instead the model receives an input value of one for an active cell as if it responds to a preferred stimulus, or zero as a non-preferred input. It will then receive equally distributed random input for mimicking cells of the background. The ratio of those three cell conditions could be individually adjusted. By that we are able to analyze problems when using the model more effectively and calculate the outcome of the model when receiving an optimal input.

3.5 Changing the postsynaptic threshold θ^{HVA}

The postsynaptic threshold θ^{HVA} is typically chosen equal to the mean response of all HVA-cell neurons (population mean). In a larger population of cells, this value is very low as the HVA-cells tend to respond sparse to a given input, and thereby most of the neurons will not fire. When the threshold is too low the cells will learn not only their preferred input, but also an input with close to similar but not equal properties e.g. a wide range of orientations.

Temporal population-based θ^{HVA} Because the maximum activity of the HVA-cells is fluctuating in a wide range, one may instead use the temporal population-based mean activity as a threshold. This threshold is calculated incrementally using:

$$\theta^{\mathrm{HVA}}(t) = \theta^{\mathrm{HVA}}(t-1) + \frac{1}{t} \cdot \Big(< r^{\mathrm{HVA}}(t) > - \theta^{\mathrm{HVA}}(t-1) \Big) \tag{2}$$

Constant threshold θ^{HVA} The threshold is set to a constant value, e.g. 0.5, and does not change within the learning, producing a better transferability to a complete model with more cells, due to independence from the number of V1-neurons. It works fine as long as the threshold is not higher then the overall activity.

4 Results

The novel learning algorithm normalizes the learning speed based on two components: a faster learning of weights towards zero and a balancing of the learning speed for weights with positive or negative weight changes.

4.1 Faster weight changes towards zero

When learning on real world images, the weights of cells responding to the background equilibrate at a value that is not zero, due to a difference of responses in the input dataset. Therefore we need to compensate this error by a separate scaling of weights changing towards zero in comparison to those developing away from zero.

$$\tau_w = \begin{cases} \frac{\tau_{FF}}{\zeta} & , W \to 0 \\ \tau_{FF} & , else \end{cases} \tag{3}$$

Whenever a cell changes its weight in the direction of zero, it will learn faster with a factor of the chosen parameter ζ. We have chosen a value of 2.5 after several tests. With that value, the weights to background-cells are running toward zero (see Fig. 3), resulting in a better noise-suppression as well as a good figure ground separation.

4.2 Balancing of the learning rate

When applying a presynaptic threshold for the input neurons, the response of a given input gets shifted. This was tested on a normally distributed input. Based on the height of the threshold, the learning speed should be adapted as follows to prevent a shift towards positive or negative weights.

$$\tau_{w,new} = \tau_{w,old} \cdot \tau_{pre}, \quad where: \tau_{pre} = \begin{cases} \frac{1}{1-\theta_{V1}} & , \Delta W > 0 \\ \frac{1}{\theta_{V1}} & , \Delta W < 0 \\ 1 & , \Delta W = 0 \end{cases} \tag{4}$$

These two modifications result in a lower response to non-specific V1-cells and thereby a better noise-resistance of the HVA-layer, as visible in Figure 3.

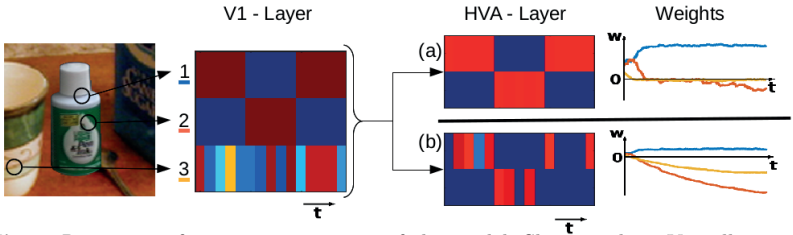


Fig. 3. Responses of a miniature version of the model. Showing three V1 cells, two HVA cells and the synaptic weights towards the first (upper) HVA cell for the complete learning time. a) With normalization of τ the HVA responses match the V1-input, b) without normalization the cells do not respond correctly because they get inhibited by the activity of the background cell(3, yellow). We used the values: $\alpha = 400$, $\tau_{FF} = 285$

5 Conclusion and future work

With the given modifications, the model is now able to learn background-invariant object representations in a miniature model. They show a good suppression of unpredictable responses of the V1-cells. In a complete model, the system still tends to learn slightly negative weights for background positions. We ascribe this problem putatively to the different number of V1 and HVA cells between both models. Several of the models parameters depend on the number of neurons, thus a finer tuning of the parameters is necessary. We are currently conducting a study solving these issues. Nevertheless, the current study shows already how the human brain may operate to achieve figure-ground separation and learn background-invariant object representations.

References

1. Antonelli, M., Gibaldi, A., Beuth, F., Duran, A.J., Canessa, A., Chessa, M., Solari, F., Del Pobil, A.P., Hamker, F., Chinellato, E., et al.: A hierarchical system for a distributed representation of the peripersonal space of a humanoid robot. Autonomous Mental Development, IEEE Transactions on 6(4), 259–273 (2014)
2. Beuth, F., Hamker, F.H.: Attention as cognitive, holistic control of the visual system (2015)
3. Chikkerur, S., Serre, T., Tan, C., Poggio, T.: What and where: A bayesian inference theory of attention. Vision research 50(22), 2233–2247 (2010)
4. Földiák, P.: Learning Invariance from Transformation Sequences. Neural Computation 3(2), 194–200 (1991)
5. Goerick, C., Wersing, H., Mikhailova, I., Dunn, M.: Peripersonal space and object recognition for humanoids. In: Humanoid Robots, 2005 5th IEEE-RAS International Conference on. pp. 387–392. IEEE (2005)
6. Hamker, F.H.: The reentry hypothesis: the putative interaction of the frontal eye field, ventrolateral prefrontal cortex, and areas V4, IT for attention and eye movement. Cerebral Cortex 15(4), 431–447 (2005)
7. Hubel, D.H., Wiesel, T.N.: Receptive fields of single neurones in the cat's striate cortex. The Journal of Physiology 148(3), 574–591 (1959)
8. Nene, S.A., Nayar, S.K., Murase, H.: "Columbia Object Image Library (COIL-100)" (1996)
9. Raudies, F., Neumann, H.: A neural model of the temporal dynamics of figure-ground segregation in motion perception. Neural Netw 23, 160–176 (2010)
10. Spratling, M.W.: Learning viewpoint invariant perceptual representations from cluttered images. IEEE 27(5), 753–761 (2005)
11. Walther, D.B., Koch, C.: Attention in hierarchical models of object recognition. Progress in brain research 165, 57–78 (2007)

Distributed Event-based Communication for Web Components

Michael Krug

Technische Universität Chemnitz, Germany
michael.krug@informatik.tu-chemnitz.de

Abstract. The set of new W3C standards called Web Components brings reuse-oriented, component-based application development to the Web. By providing custom element names, encapsulation and interoperability, developers are able to create components that can be reused in any HTML5-based web application. Polymer is a framework by Google Inc. that utilizes the opportunities of Web Components and hides the comprehensive set of technologies behind a declarative definition and adds additional features. Unfortunately, inter-component communication, which is required for application development by composition, is not covered by neither the W3C specifications nor the Polymer project. Therefore, we propose an approach to enable event-based communication for Web Components. We facilitate loosely coupling and support single- as well as multi-screen usage scenarios by providing publish/subscribe mechanisms and message distribution across multiple devices. To showcase the applicability of our approach, we present an extension to the Polymer framework as an exemplary implementation.

1 Introduction

Web Components[1] are defined by set of standards currently being produced as W3C specifications that facilitate the creation of reusable components in web applications. Web Components are supposed to bring component-based software engineering to the Web by providing encapsulation and interoperability of individual HTML elements. They enable developers to define components with a high level of visual richness and interactivity not possible with CSS alone, and ease of composition and reuse not possible with script libraries today. The set consists of the following four new standards: *Templates*, *Custom Elements*, *Shadow DOM* and *HTML Imports*. With Polymer[2], a framework was introduced that deploys those new technologies to provide a simple way of creating new Web Components using a declarative syntax. Its aim is to reduce the amount of code needed to build complex components for the Web. Web Components provide a modern and promising way to create encapsulated, programmable and reusable components. To fully enable component-based web engineering, the

[1] http://w3c.github.io/webcomponents/
[2] http://www.polymer-project.org

aspect of loosely coupling of components needs to be considered. With regard to application development by composition, components need to communicate with each other via interfaces. This functionality is not yet covered by neither the W3C specifications nor the Polymer project. Therefore, we propose an approach to enable inter-component and inter-device communication for Web Components. This will facilitate the development of web applications by composing components that can exchange data. Additionally, we add support for multi-device applications by providing a service for distributing messages across multiple connected devices (devices with the same endpoint that share the same context). Example target domains could be dashboard or mashup like applications that can be used concurrently on e.g., desktop and mobile devices.

Similar approaches are, e.g., Dojo widgets[3], W3C Widgets[4] and OpenSocial Widgets[5]. Dijit is a UI library, which provides a widget system layered on top of Dojo. Dojo widgets are configured using attributes in the HTML markup and can utilize a topic-based publish/subscribe mechanism for communication purposes. W3C Widgets (also called Packaged Web Apps) and OpenSocial Widgets are open web standards. They need to be executed in special platform environments, such as Apache Rave or Apache Shindig. The integration of OpenAjax Hub into Apache Rave described in [3] is an approach to achieve communication between those widgets. The DireWolf framework [1] integrates multi-device communication into the Apache Shindig platform.

In the following, we present our approach to facilitate communication for Web Components, which aims to enable the development of single- and multi-device web applications by composition.

2 Approach

Loosely coupling of components is important to ensure reusability and enable new compositions. To support message exchange between Web Components we are using the ideas presented in [2] by Krug et.al. and propose an event-driven communication channel using a topic-based publish/subscribe mechanism. By applying the publish/subscribe pattern, components do not need to know about other communication partners, which ensures loosely coupling. Thus, every component can work regardless of others. We propose a communication API for Web Components that provides methods for: *subscribing to a topic*, *canceling a subscription*, *publishing a message with a given topic* and *receiving a message for subscribed topics*. Every component can act as a publisher as well as a subscriber and can subscribe to any number of topics. The topic (or logical channel) is described by a string value. When subscribing to a topic a callback handler needs to be specified. This enables the message bus to actively inform the component about new messages on topics they have subscribed to. We suggest to use

[3] `http://dojotoolkit.org`
[4] `http://www.w3.org/TR/widgets/`
[5] `http://opensocial.atlassian.net/wiki/display/OSD/Specs`

JavaScript's native *DOM events*[6] to implement the message bus. This ensures high performant and low latency communication without blocking the user interface.

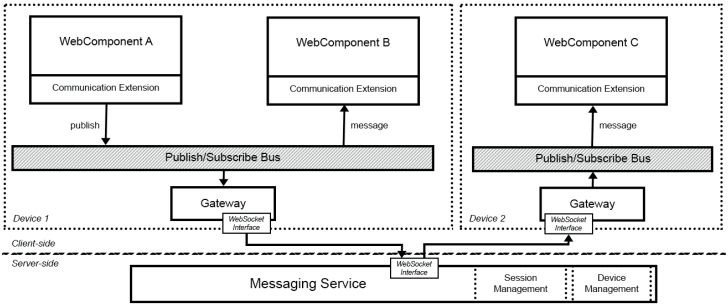


Fig. 1. Inter-component and inter-device communication architecture

Not limiting communication between components to a single device context, Krug et al. proposed a standalone WebSocket-based service to distribute messages across different devices [2]. We are using that approach to distribute the local publish/subscribe messages to other devices and enable multi-device scenarios (cf. Figure 1). On the client side a *gateway* component transparently captures all messages that are published on the bus and transmits them to the *messaging service* using the *WebSocket protocol*[7]. The *messaging service* analyzes the messages and distributes them selectively to connected devices. The service also provides device and session management functionalities for identifying and grouping devices. When the gateway component receives data from the messaging service, messages are extracted and again published to the local communication bus and can be consumed by the components. Since the gateway is working like a hook, all messages transmitted on the event bus are captured without changing the code or configuration of existing components. In the following section, we describe the implementation of our approach as an extension of the Polymer framework.

3 Extension of the Polymer Framework

By exploiting Polymer's structure of modularized APIs, we add the proposed communication API to the framework (source code repository available at: `https://github.com/michikrug/SmartComposition`). We add a set of new methods and properties to the base Polymer component object. By inserting function calls to already existing lifecycle callbacks, we inject our communication functionality in every Polymer component. We extend the following callbacks: *createdCallback*, *attachedCallback* and *detachedCallback*. These three functions are low-level lifecycle methods that are called as a part of the standard *Custom Elements*[8] operation. The *createdCallback* is invoked after the custom element

[6] `https://dom.spec.whatwg.org/#events`
[7] `https://tools.ietf.org/html/rfc6455`
[8] `http://www.w3.org/TR/custom-elements/`

instance is created and its definition is registered. We extend that callback by injecting our initialization method, which sets up an ID property to identify components, prepares a subscription object to store topics and handlers and creates a message handler to preprocess captured message events. The two other methods - *attachedCallback* and *detachedCallback* - are called whenever the custom element is inserted/removed into/from a document. We exploit these callbacks to inject the binding of our message event handlers. To implement a topic-based publish/subscribe messaging pattern, we provide a *subscribe(topic, handler)* method that requires a topic name as a string and a handler function. Both the topic and the handler get stored and a new subscription is set up. In reverse, by calling the *unsubscribe(topic, handler)* method, the handler is removed from the subscription list of the topic. The *publish(topic, data)* method provides a wrapper for dispatching custom events to enable the *publish* functionality. Those custom events are described by the given topic and the provided data, and are created and dispatched using JavaScript's native *DOM events API*.

An example application that is composed using several Web Components and uses the extension of the Polymer framework to enable inter-component as well as inter-device communication can be explored visiting the following URL: `http://vsr-demo.informatik.tu-chemnitz.de/polymer`

4 Conclusion

In this paper we presented our approach for extending Web Components with inter-component as well as inter-device communication features. Our extension of adding an event-based communication channel as well as providing a WebSocket-based messaging service enables the composition of rich web applications for usage across different platforms and multiple devices. By providing an exemplary implementation as an extension of the Polymer framework we showcase the applicability of our approach. Furthermore, developers who are already working with Polymer as well as developers who are new to those technologies are now able to create communicative Web Components. The focus of our future research will be providing a repository for reusable components and describing communication interfaces to support hassle-free composition.

References

1. Kovachev, D., Renzel, D., Nicolaescu, P., Klamma, R.: DireWolf - Distributing and Migrating User Interfaces for Widget-Based Web Applications. In: Web Engineering, pp. 99–113. LNCS, Springer Berlin Heidelberg (2013)
2. Krug, M., Wiedemann, F., Gaedke, M.: SmartComposition: A Component-Based Approach for Creating Multi-screen Mashups. In: Web Engineering, pp. 236–253. LNCS, Springer International Publishing (2014)
3. Wilson, S., Daniel, F., Jugel, U., Soi, S.: Orchestrated User Interface Mashups Using W3C Widgets. In: Current Trends in Web Engineering, pp. 49–61. LNCS, Springer Berlin Heidelberg (2012)

COP: Improving Collaboration in e-Health Utilizing the Cloud

Fabian Wiedemann

Technische Universität Chemnitz, Germany
{firstname.lastname}@informatik.tu-chemnitz.de

Abstract. While cloud computing has improved many workflows that deal with collaboration, the cloud is rarely used in the sector of e-Health applications. Therefore, we propose a scenario that can benefit from an introduction of cloud computing in e-Health. Based on this scenario we identified several challenges a system needs to deal with, such as protection of patient data records against access by third parties. In this paper we present our planned research to securely store patient data records in the cloud. We propose a first approach to deal with the identified requirements and describe our research methodology.

1 Introduction

The workflow of collaboration between patients and doctors in Germany has not changed in the last couple of years. While a lot of other sectors benefits from introducing cloud technologies, such as EDP, the health sector is mostly using desktop stand-alone applications. In this context, we identified a common workflow that tackles the documentation of blood values of patients with diabetes. For example, these patients need to measure multiple times a day their blood sugar level. The values are written in a journal which is shown to the doctor in a monthly or quarterly cycle. Based on these values the doctor can change the medication.

While the doctor can only react on changed blood values when the patient comes to her office, the doctor cannot react immediately when blood values change dramatically. To improve this workflow the cloud can be leveraged to immediately send blood values from the patient's home to the doctor's office. Thus, the recorded values can be automatically added to the patient chart. When the doctor discovers something abnormally she can immediately contact the patient to order her to the doctor's office.

One big issue why the cloud is currently not used in the e-Health sector is the fear of unauthorized access to sensitive data [5]. Furthermore, the issue of privacy for the patient's e-Health records in Germany is regulated by a lot of laws and directives. To deal with these laws and directives special requirements need to be fulfilled by the software and cloud infrastructure. Thus, it is the purpose of my PhD to remove the obstacles which hinder the usage of the cloud for e-Health software.

2 Research Challenges

To deal with the scenario described in Section 1 we analyzed the following research challenges our approach needs to fulfill. When storing sensitive data in the cloud, the major issue is the protection of the data against third parties. That is, some attackers who might get access in an illegal way as well as service engineers or system administrators of the cloud provider should not be able to access the plain text data. Furthermore, the access of data by the end users, such as doctors or patients, should be performed in an adequate time. While there are stored many data records of patients in such a cloud system, the cloud system should perform search and processing of the data fast to not mar the user experience. Because in such a scenario different persons have different roles which imply different access rights on the data, a cloud system for e-Health records should offer a fine-grained identity management and access control.

3 Related Work

Literature to encryption partially deals with search-able encryption schemes. Bellare et al. in [1] and Song et al. in [3] focus on an encryption scheme which is secure against common attacks, while offering an efficient search on encrypted keywords. They propose an approach of a public-key encryption scheme that provides a deterministic encryption algorithm. That is, encrypting the same plain text, for example a keyword, two times with the same encryption key results in the same cipher text. Thus, searching on the encrypted keywords will provide the same results as searching on the plain keywords.

Besides the security topic in this research there is also related work in the context of software engineering. In [4] Vitali et al. describe an approach to build an e-Health infrastructure on the web. While our approach focuses on a client side encryption and storing only encrypted data, Vitali et al. developed a RESTful service to communicate e-Health records between medical practices, patients, and hospitals. Based on their recent work they evolved the framework SOLE to a REST architecture to reduce the overhead required by the previous one that uses SOAP for transmitting messages.

4 Approach

To deal with the challenges described in Section 2, we propose the following two-stage approach for secure storing patient data records in the cloud. One stage realizes the secure storing of the patient data records in the cloud (cf. Figure 1 (a)), while the other stage utilizes the requesting of the patient data records from the cloud (cf. Figure 1 (b)). Our approach works as follows.

First, the client extracts keywords from the patient data record that should be stored in the cloud (cf. ①). As keywords we understand anything that is related to identify the patient data record, such as the patient's names. Afterwards, the client encrypts the patient data record and the keyword-file-table (cf. ②). The

key used for encryption is only stored on the client. Thus they key has not to be revealed to the framework or the cloud. Both, the encrypted patient data record and the encrypted keyword-file-table are sent to the framework (cf. ③). The framework stores the encrypted keyword-file-table for an efficient lookup performed during a search request. The encrypted patient data record will be stored in the cloud (cf. ④).

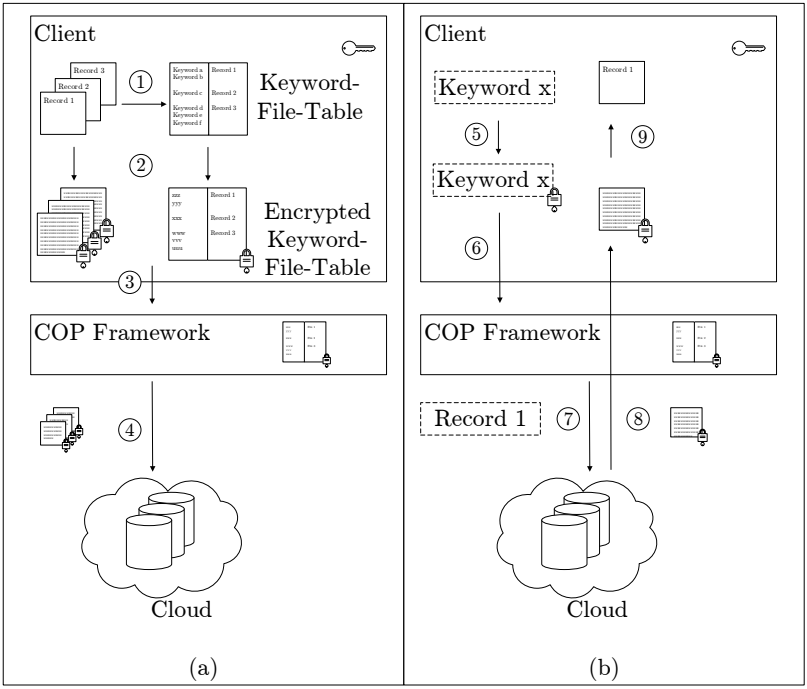


Fig. 1. Architecture of the COP approach.

When the client wants to request a patient data record to a specific keyword, the client encrypts the keyword using its stored key (cf. ⑤). The encrypted keyword is sent to the framework (cf. ⑥) and the framework can compute which patient data record is requested based on the encrypted keyword-file-table. Afterwards, the framework requests the encrypted patient data record from the cloud (cf. ⑦). The framework receives the encrypted patient data record from the cloud and sends it to the client (cf. ⑧). In the last step, the client decrypts the encrypted patient data record with its stored key and can process the patient data record (cf. ⑨).

5 The Research Methodology

The research proposed in this paper will be structured following the Logical Framework Approach (LFA) [2]. LFA covers three stages of our research, i.e., identification, formulation, and evaluation. In the first stage, we identified the stakeholders of the proposed problem, i.e., doctors, nurses, patients, developers, and cloud providers. By interviewing and surveying these stakeholders we want to capture and analyze their problems. Based on the analyzed problems we derive objectives our approach needs to achieve. Within the formulation stage, we create measurable outcomes and define the scope of our research. Independently from LFA, we implement our approach between formulation and evaluation stage. In the last stage, we evaluate and measure how our objectives are fulfilled.

6 Conclusions

This paper presents an approach for a framework to securely store patient data records in the cloud. Based on a use case in the e-Health sector we analyze problems and challenges for the success of the proposed approach. We describe a first draft how we want to tackle the problems of a full client-side encryption and the challenge of a search-able encryption scheme.

Future work will focus on fine-grained access control. That is, maybe another doctor should get access to special parts of one patient data record. Therefore, only the shared parts of the patient data record should be revealed to this other doctor.

Bibliography

[1] Bellare, M., Boldyreva, A., O'Neill, A.: Deterministic and efficiently searchable encryption. In: Menezes, A. (ed.) Advances in Cryptology - CRYPTO 2007, Lecture Notes in Computer Science, vol. 4622, pp. 535–552. Springer Berlin Heidelberg (2007)

[2] European Commission: Project Cycle Management Guidelines, volume 1 edn. (2004), `http://ec.europa.eu/europeaid/multimedia/publications/documents/tools/europeaid_adm_pcm_guidelines_2004_en.pdf`

[3] Song, D.X., Wagner, D., Perrig, A.: Practical techniques for searches on encrypted data. In: 2000 IEEE Symposium on Security and Privacy (S&P 2000). pp. 44–55. IEEE (2000)

[4] Vitali, F., Amoroso, A., Roccetti, M., Marfia, G.: Restful services for an innovative e-health infrastructure: A real case study. In: 2014 IEEE 16th International Conference on e-Health Networking, Applications and Services (Healthcom). pp. 188–193. IEEE (2014)

[5] Wallraf, B.: Cloud-monitor 2014. Tech. rep., KPMG AG (2014)

CloudLinkSync: URI-basierte Adressierung lokaler Kopien von Cloud-Ressourcen

Frank Burian

Fakultät für Informatik
Verteilte und Selbstorganisierende Rechnersysteme (VSR)

frank.burian@informatik.tu-chemnitz.de

Abstrakt. Die Cloud erlangt zunehmend an Bedeutung, wenn es um die Kollaboration zwischen Teams und Unternehmen geht. Wichtige Ressourcen, wie Dokumente, können gespeichert und mit Mitarbeitern geteilt werden, welche durch die entsprechenden Synchronisations-Tools auch als lokale Kopien auf den einzelnen Clients speicherbar sind. Da diese Kopien der Ressourcen auf den Clients oft unterschiedliche physikalische Adressen besitzen, können diese nicht direkt über einen eindeutigen Link zugänglich gemacht werden. CloudLinkSync löst dieses Problem und ermöglicht eine logische Adressierung auf diese unterschiedlichen verteilten Kopien mittels frei definierbaren URI-basierten Adressierungen.

1 Einleitung

Das Speichern und Teilen von Ressourcen innerhalb der Cloud erlangt in Teams und Unternehmen heutzutage zunehmend an Bedeutung. Solche Ressourcen können u. a. Dateien wie Dokumente und Medien sein. Cloud-Dienste wie Dropbox[1], Google Drive[2], OneDrive[3] oder OwnCloud[4] ermöglichen es, solche Ressourcen im Internet zentral zu speichern und über zur Verfügung stehende Synchronisations-Tools auf mehrere physikalische Geräte (Clients) zu verteilen. Somit wird die Zugänglichkeit dieser Ressourcen zu jeder Zeit und von jedem Client aus gewährleistet. Diese Cloud-Dienste erlauben zudem auch das Teilen bzw. Freigeben von Ressourcen mit anderen Nutzern, was die Kollaboration in Teams und Unternehmen ermöglicht. So können diese mittels der verschiedenen Cloud-Dienste ihre Ressourcen zentral speichern und lokale Kopien auf die einzelnen Clients der Mitarbeiter verteilen.

[1] https://www.dropbox.com
[2] https://drive.google.com
[3] https://onedrive.live.com
[4] https://owncloud.org

2 Problemstellung & Ziel

Oft ist es erforderlich die Mitarbeiter über die Existenz einer neuen oder bereits vorhandenen Ressource zu informieren. Dies geschieht oft auf elektronischem Wege, z. B. per E-Mail oder Instant Messaging. In diesem Kontext wäre es wünschenswert, die Mitarbeiter über eine direkte logische Adressierung, z. B. einen Link/Verweis, auf die entsprechenden lokalen Kopien weiterzuleiten. Diese Art der logischen Adressierung lokaler Kopien von Cloud-Ressourcen ist allerdings nicht immer möglich, da diese auf den einzelnen Clients in der Regel unterschiedliche physikalische Adressierungen (Dateipfade) besitzen. CloudLinkSync soll dieses Problem lösen und eine Abbildung von physikalischen Ressourcen durch eine logische Adressierung ermöglichen. Diese Adressierungen sollen für den Anwender frei wählbar und für jede Ressource (Dateityp) gelten. Der Zugriff auf die physikalischen Ressourcen soll zudem direkt und ohne große Verzögerungen erfolgen, sowie plattformübergreifend für die Betriebssysteme Windows, Linux und MacOS anwendbar sein.

3 Related work

Für eine logische Adressierung von physikalischen Ressourcen existieren bereits einige Ansätze bzw. Technologien. Das Domain Name System (DNS) ermöglicht z. B. die Abbildung physikalischer Ressource über eine Domain bzw. URL, welche in der Regel auch frei wählbar sind. Allerdings adressiert das DNS nur IP-Adressen von Servern und keine physikalischen Ressourcen in Form von Dateien [1]. Mittels Linked Data lassen sich frei verfügbare Ressourcen, hauptsächlich Informationen und Daten, über einen Uniform Resource Identifier (URI) logisch adressieren. Da die URI von Linked Data auf dem HTTP-Protokoll basieren, können allerdings nur Ressourcen adressiert werden, welche sich auf Webservern befinden [2]. Ein weiterer Ansatz sind die Uniform Resource Names (URN). Die URN ist ein URI mit einem festdefinierten URI-Schema »urn« und dient als eindeutiger, dauerhafter und ortsunabhängiger Bezeichner für eine beliebige Ressource, adressiert diese aber nicht [3] [4].

Zusammengefasst ermöglicht keiner dieser Ansätze bzw. Technologien die Lösung des genannten Problems, da keiner dieser eine logische Adressierung einer auf dem Client lokal befindlichen Ressource ermöglicht.

4 CloudLinkSync

Um das beschriebene Problem zu lösen nutzt die Lösung CloudLinkSync, URI-Schemata, welche in den genannten Betriebssystemen frei definierbar sind. In Verbindung mit den Wurzelverzeichnissen der einzelnen Cloud-Dienste und weiteren definierbaren Unterverzeichnissen, verweist CloudLinkSync über eine entsprechende URI auf die jeweiligen Ressourcen, auch wenn diese auf den einzelnen Clients unterschiedliche physikalische Adressierungen besitzen.

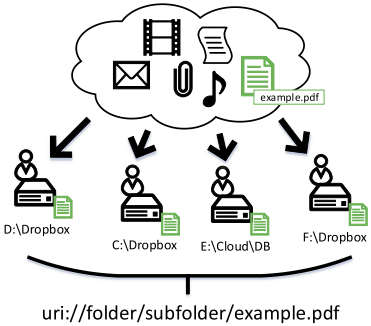


Abbildung 1: Ansatz & Funktionsarchitektur von CloudLinkSync

Um die Wurzelverzeichnisse der einzelnen Cloud-Dienste, sowie die zu verwendende URI-Schemata mit deren Ausgangsverzeichnissen zu kennen, bedient sich die Lösung zweier Konfigurationen, der Provider- & URI-Schemata-Konfiguration. Diese Konfigurationen basieren in der softwaretechnischen Umsetzung auf dem XML-Format.

```
<cloudlinksync>
    <providers>
        <provider id="dropbox">d:\dropbox\</provider>
        <provider id="googledrive">d:\google drive\</provider>
        <provider id="owncloud">d:\owncloud\</provider>
    </providers>
</cloudlinksync>
```

Auflistung 1: Provider-Konfiguration

Die Provider-Konfiguration, welche pro Client individuell definiert wird, beinhaltet die lokalen Wurzelverzeichnisse der verschiedenen Cloud-Dienste, unter welchen die entsprechenden Kopien der in der Cloud befindlichen Ressourcen gespeichert werden.

```
<cloudlinksync>
    <schemes>
        <scheme name="comp01">
            <locations group="depA">
                  <folder provider="dropbox">\documents\</folder>
                  <folder provider="googledrive">\</folder>
            </locations>
            <locations group="depB">
                  <folder provider="owncloud">\</folder>
            </locations>
        </scheme>
    </schemes>
</cloudlinksync>
```

Auflistung 2: URL-Schemata-Konfiguration

Die URI-Schemata-Konfiguration definiert, welche URI-Schemata im Betriebssystem verankert werden und verknüpft die dazugehörigen Ausgangsverzeichnisse mit den in der Provider-Konfiguration festgelegten Wurzelverzeichnissen.

Wird beispielsweise auf einem Client die URI »comp01://depA/example.pdf« aufgerufen, löst CloudLinkSync entsprechend der beschriebenen Konfigurationen, die URI auf den Pfad »d:\dropbox\documents\example.pdf« auf und prüft die Existenz dieser Ressource. Sollte diese nicht existieren, wurde unter dem XML-Knoten »locations« mit dem dazugehörigen Attribut »group="depA"« ein weiteres Ausgangsverzeichnis definiert, was im nächsten Schritt auf die Ressource »d:\google drive\example.pdf« versucht zu verweisen. Die Reihenfolge der definierten Ausgangsverzeichnisse durch das XML-Tag »folder« spiegelt somit die Priorisierung der Aufrufe wieder. Dies ist sinnvoll, wenn z. B. Ressourcen nur eine bestimmte Lebensdauer in einem Verzeichnis besitzen und anschließend in ein anderes Verzeichnis zur Archivierung verschoben werden.

5 Zusammenfassung

Dieses Paper zeigt mit CloudLinkSync eine Lösung, um über frei definierbare URI eine logische Adressierung von lokalen physikalischen Ressourcen, auch wenn diese auf den einzelnen Clients unterschiedliche Speicherorte besitzen, zu ermöglichen. Dabei ist eine logische Adressierung aus hierarischer Sicht auch auf mehrere physikalische Ressourcen möglich. Eine prototypische Implementierung des beschriebenen Ansatzes wurde bereits erfolgreich umgesetzt. Zukünftig könnte die Provider-Konfiguration ggf. durch eine automatische Erkennung der Wurzelverzeichnisse der Cloud-Dienste innerhalb der Softwarelösung ersetzt werden, da diese in jedem Betriebssystem ermittelbar sind. Dies schränkt allerdings die Verwendbarkeit der Lösung auf die einzelnen Cloud-Dienste ein, da die XML-Konfiguration auch einen Einsatz außerhalb der Cloud-Dienste ermöglicht.

Literaturverzeichnis

1. Mockapetris, P.: Domain Names - Concepts and Facilities, https://tools.ietf.org/html/rfc1034.
2. Berners-Lee, T.: Linked Data, http://www.w3.org/DesignIssues/LinkedData.html.
3. Moats, R.: URN Syntax, https://www.ietf.org/rfc/rfc2141.txt.
4. Schöning-Walter, C.: Der Uniform Resource Name (URN). (2009).

Generisches Framework zur Parallelisierung von Echtzeitfähiger Bilddatenauswertung für Rekonfigurierbare Integrierte Schaltkreise

Stephan Blokzyl und Wolfram Hardt

Technische Universität Chemnitz
Fakultät für Informatik, Lehrstuhl Technische Informatik
Straße der Nationen 62, D-09111 Chemnitz, Germany
sblo@hrz.tu-chemnitz.de, hardt@hrz.tu-chemnitz.de
http://www.tu-chemnitz.de/cs/ce

Die Evolution bilddatengestützter Mess- und Detektionssysteme wird kontinuierlich durch verschiedene Technologiebereiche getrieben. Hardwareinnovationen stellen immer leistungsfähigere Sensoren und Verarbeitungseinheiten bereit. Neue Methoden verbessern stetig Sensordatenakquise und -signalauswertung, und die zunehmende Vernetzung und Interaktion unterschiedlicher Verarbeitungsarchitekturen steigern Heterogenität und Interoperabilität moderner Bildauswertungssysteme [4]. Diese Entwicklungen führen neben einer allgemeinen Leistungssteigerung auch zu neuen Herausforderungen. So erzeugen leistungsfähigere Sensoren mit zunehmender Auflösung größere Sensordatenvolumina, die im Verarbeitungssystem verlustfrei, mit hoher Qualität aufgezeichnet, verwaltet und ausgewertet werden müssen. Diese Verarbeitungsschritte stellen insbesondere bei Automotive-, Schienenverkehrs-, Luft- und Raumfahrtanwendungen eine große Herausforderung dar, in denen Sensorsignalauswertung mithilfe von eingebetteten Systemen realisiert ist und Aspekte wie Zuverlässigkeit sowie Echtzeitfähigkeit eine zentrale Rolle spielen [1].
Eine Lösung bieten anwenderprogrammierbare, integrierte Schaltkreise (Field Programmable Gate Arrays, FPGAs). Sie vereinen Parallelisierungsfähigkeit und Verarbeitungsgeschwindigkeit von integrierten Schaltkreisen mit der Flexibilität, Skalierbarkeit und Elastizität softwarebasierter Verarbeitungssysteme. Ferner erreichen FPGAs eine hohe Verarbeitungsleistung aufgrund optimaler Anpassung an die Zielapplikation und bieten eine ausgewogene Balance zwischen Leistungsfähigkeit, Effizienz und Parallelisierbarkeit. Diese Eigenschaften bilden die Grundlage des von Blokzyl *et al.* eingeführten Konzepts zur Hardwarebeschleunigung für Echtzeitbildauswertung mithilfe von Daten- und Funktionenparallelisierung [2]. Ziele dieser zweidimensionalen Parallelisierungsmethodik (Daten/Funktion) sind die Beschleunigung des Bildauswertungsprozesses bei gleichzeitiger Erhöhung der Verarbeitungsgüte, und die Umsetzung spezifischer Anforderungen an Systemkomponenten für sicherheitskritische Perzeptionsanwendungen:

- Flexibilität durch Modularisierung und Rekonfigurierbarkeit
- Hohes Beschleunigungspotential durch parallele Datenverarbeitung
- Echtzeitfähigkeit durch Bestimmbarkeit der maximalen Ausführungszeit

- Ergebnistreue durch Einsatz deterministischer Algorithmen
- Sicherheit durch Güte-/Vertrauensbewertung der Verarbeitungsergebnisse
- Zertifizierbarkeit der Systeme für sicherheitskritische Einsatzszenarien

Für die leistungsfähige FPGA-basierte Umsetzung der zweidimensionalen Parallelisierungsmethodik wird ein On-Chip-Framework benötigt, das neben Flexibilität und Skalierbarkeit vor allem ein hohes Maß an Parallelisierung originär unterstützt. Zu diesem Zweck bietet die im Folgenden vorgestellte PARIS[1]-Systemarchitektur einen Lösungsansatz, der die hochgradig parallele Integration einer Vielzahl von Daten- und Funktionsparallelisierungsmodulen für Sensordatenverarbeitung und -fusion, sowie der Steuerung des Gesamtsystems, des Datenflusses und der angeschlossenen Sensoren erlaubt (siehe Abb. 1). Sie teilt sich in folgende zentrale Systemkomponenten auf: Netzwerkanbindung und -management, System- und Sensormanagement sowie Bilddatenauswertung und Fusion. In den folgenden Abschnitten werden die einzelnen Systemelemente vorgestellt und ihre Funktionsweise näher erläutert.

Netzwerkanbindung und Netzwerkmanagement umfassen sowohl die physikalische als auch die logische Anbindung der PARIS-Architektur an das Gesamtsystemumfeld. Der für die Kommunikation verwendete Gigabit-Ethernet-Standard (IEEE 802.3) ist aufgrund der freien Konfigurierbarkeit eines FPGA beliebig durch andere Datenübertragungsstandards ersetzbar. Jedoch bieten moderne DSP-FPGAs (z.B. Xilinx Virtex- oder Altera Stratix-Familien) fest verdrahtete Systemkomponenten zur Medienzugriffsteuerung für Gigabit-Ethernet. Diese können ohne Implementierungsmehraufwand direkt verwendet werden und es müssen nur Schichten mit höherem Abstraktionsgrad (ab Layer 3, vgl. Open Systems Interconnection Model) auf dem FPGA realisiert werden. Alle Layer 1-Funktionen (el. Nachrichtenübertragung) übernimmt ein externer Transceiver. Die Kommunikation zwischen PARIS-Framework und Transceiver verwendet das standardisierte Gigabit Media Independent Interface (GMII), welches unabhängig vom eingesetzten Kommunikationsmedium ist. Zwei eigenständige Interfaces stellen dem Gesamtsystem eine Kanalkapazität von insgesamt 2 Gbit/s für Sensordatenakquise und Ergebnisverteilung zur Verfügung. Mehrere First-In-First-Out-Eingangspuffer (FIFOs) stehen bereit, um große Bildpakete mit bis zu 9KB Payload pro Frame (Jumboframes) parallel empfangen und verwerten zu können. Damit sinkt das Risiko, dass Warte- bzw. Gültigkeitszeiten zeitkritischer Kommunikation verfehlt oder ankommende Pakete verworfen werden.

System- und Sensormanagement Aufgabe des System- und Sensormanagements ist die Weiterverteilung der empfangenen Pakete zu den zugeordneten Verarbeitungsmodulen. Werden z.B. vier Empfangspuffer für den Datenempfang eingesetzt (vgl. Abb. 1), dauert es bei einem Systemtakt von 125MHz höchstens 24ns, bis eine vollständig empfangene Nachricht erkannt und in Abhängigkeit ihrer Nachrichtenklasse weitergeleitet wird. Ein ARP ICMP IGMP Controller setzt die OSI-Layer für Sicherung und Vermittlung um und realisiert das gesamte

[1] **P**arallelisation **A**rchitecture for **R**eal-time **I**mage data exploitation and **S**ensor data fusion

Ethernet- sowie Kommunikationsmanagement. Ein Sensor Controller steuert auf Basis des GigE Vision Protokolls alle angeschlossenen elektrooptischen Sensoren. Ein PARIS Controller überwacht die Funktion des gesamten Verarbeitungsframeworks, verwaltet alle Systemparameter, regelt und steuert die unterschiedlichen Arbeitsmodi. Das Kernelement der PARIS-Architektur bildet der Image Processing Controller. Er beherbergt alle Verarbeitungseinheiten zur Sensorsignalauswertung sowie Fusion und verfolgt zweidimensionale Parallelisierungsstrategien.

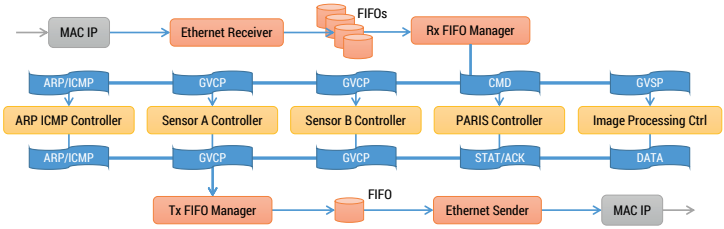


Abbildung 1. Empfänger-, Sender- und Controllerarchitektur des PARIS-Frameworks

Alle Verarbeitungsergebnisse der einzelnen Systemmodule werden in Ethernetframes verpackt und in einem FIFO für den Versand abgelegt. Den Pufferzugriff steuert ein Tx FIFO Manager, der mithilfe eines Tokensystems konkurrierenden Modulen Schreibzugriff erlaubt. Der Sende-FIFO ist als singulärer Puffer für jede Ethernet-Schnittstelle ausgeführt. Denn die kleinen Framegrößen bzw. -größenvarianzen der Ausgangspakete führen zu einer deutlich geringeren Zugriffskonkurrenz im Vergleich zum Empfangspuffer. Erzeugt eine Anwendung sehr (unterschiedlich) große Ausgangspakete, ist der Tx FIFO Manager um zusätzliche Pufferspeicher erweiterbar. Paralleles Senden kleiner Ethernetframes während des Pufferns großer Ausgangspakete ist dann möglich und reduziert das Risiko des gegenseitigen Blockierens von Verarbeitungsmodulen. Der Sendespeicherstatus wird kontinuierlich von einem Ethernet Sender überwacht, der vollständig gepufferte Ethernetframes an den MAC-IP-Core/Transceiver übermittelt.

Bildauswertung und Fusion Der Image Processing Controller ist das zentrale Modul der PARIS-Systemarchitektur und realisiert das Parallelisierungsframework für Echtzeitbildverarbeitung und Fusion. Dies schließt Sensordatenakquise, Signalvorverarbeitung sowie die Bilddatenauswertung ein. Der Rohdatenstrom der elektrooptischen Sensoren wird in ein Farbbild überführt und in einem Bildsynchronisationspuffer gespeichert. Sind ausreichend Bilddaten für die Auswertung eines Bildausschnittes verfügbar, starten nachfolgende, parallele Bildauswertungsmodule die Weiterverarbeitung. Ausschlaggebend ist dabei die hinreichende Verfügbarkeit von Daten für das Folgemodul mit der größten Datenforderung. Benötigt ein Folgemodul z.B. neun Pixel zum Start und ein anderes, paralleles Modul 49 Pixel, starten beide Module erst, wenn 49 Pixel im Eingangspuffer vorliegen (Startsynchronisation). Der Verzicht auf eine vollständige Zwischenspeicherung der Eingangsdaten und das verwendete Pipelining reduzieren im Ergebnis die benötigte Bildpuffergröße deutlich im Verglich zu konventio-

nellen, softwarebasierten Verarbeitungssystemen. So wird z.B. bei Verarbeitung von zwei Megapixel großen Farbbildern eine Speicherressourceneinsparung von mehr als 99% erzielt (2KB FPGA-BRAM vs. 2MB DDR-SDRAM).
Am Ausgang des Bildsynchronisationspuffers können beliebig viele Bildauswertungsmodule parallel instanziiert werden. Ihre unterschiedlichen Lauf- und Verarbeitungszeiten werden mithilfe von Ausgangspuffern synchronisiert (Ergebnissynchronisation). Dieser Aufbau erlaubt dem Parallelisierungsframework den Einsatz von Daten- und Funktionenparallelisierungsstrategien, sowie statische und dynamische In-System-Rekonfiguration. Datenparallelisierung verteilt die Menge der zu behandelnden Daten auf unabhängige, gleichartige Verarbeitungseinheiten (Beschleunigung durch Lastverteilung) und Funktionenparallelisierung verknüpft verschiedene, schwache Musterdetektoren zu einem robusten Gesamtdetektor für Strukturen höheren Abstraktionsgrads. Alle Teilergebnisse der Daten- und Funktionenparallelisierung werden in einem Fusionsschritt zusammengeführt und in ihrer Gesamtheit bewertet. Konzeptentwurf und Realisierung dieses Fusionsansatzes, einschließlich der Einbeziehung weiterer heterogener Datenquellen sind in [2] näher erläutert. Darüber hinaus erlaubt die parallele Struktur des Image Processing Controllers die statische und dynamische Rekonfiguration des Systems, ohne dass es vollständig angehalten werden muss. Zur Laufzeit kann das Framework auf veränderte Detektionsbedingungen und -aufgaben reagieren, sowie der Parallelisierungsgrad innerhalb der Verarbeitungskette angepasst werden. Dies gewährleistet jederzeit einen optimalen Kompromiss zwischen Energiebedarf, Leistungsfähigkeit und Beschleunigungsrate.

Die vorgestellte PARIS-Systemarchitektur stellt ein Framework für modulare, flexible Parallelisierung von Echtzeitbildauswertung bereit. Sie unterstützt Daten- und Funktionenparallelisierungsstrategien sowie Rekonfigurierbarkeit, unter Beachtung grundlegender Anforderungen an eingebettete Systeme (Leistungsaufnahme, Verlustleistung, Energieeffizienz, Baugröße, Gewicht etc.). Das Framework wurde auf dem flugtauglichen COTS-Hardwarebeschleunigungsboard 4DSP FM680 erfolgreich umgesetzt und soll in einem stereokamerabasierten Bildauswertungssystem zur Lokalisierung eines unbemannten Fluggerätes während der finalen Landeanflugphase erprobt werden [3].

Literatur

1. Blokzyl, S., Hardt, W.: FPGA-basierte Hardwarebeschleunigung für Echtzeitbildverarbeitung und Fusion. In: Proceedings of the Informatiksymposium Chemnitz 2012; Chemnitzer Informatik-Berichte CSR-12-01 (2012)
2. Blokzyl, S., Vodel, M., Hardt, W.: A Hardware Accelerated Real-Time Image Processing Concept for High-Resolution EO Sensors. In: Proceedings of the 61. Deutscher Luft- und Raumfahrtkongress (2013)
3. Blokzyl, S., Vodel, M., Hardt, W.: FPGA-based Approach for Runway Boundary Detection in High-resolution Colour Images. In: Proceedings of the 2014 IEEE Sensors & Applications Symposium (SAS2014) (2014)
4. Vodel, M., Blokzyl, S., Hardt, W.: Hochauflösende Videodatenanalyse in Echtzeit - FPGA Hardwarebeschleunigung durch Daten- & Task-Parallelisierung. In: Proceedings of the Innosecure 15 - Innovationen in den Sicherheitstechnologien (2015)

Big Data Applications in Wireless Sensor Networks

René Bergelt and Wolfram Hardt

Fakultät für Informatik
Technische Informatik
Technische Universität Chemnitz
D-09107 Chemnitz
{rene.bergelt|hardt}@informatik.tu-chemnitz.de

1 Introduction

Current data-processing applications are challenged with the fact that the total amount of user data to be stored or processed in common computing systems doubles every two years [7]. A direct consequence of this is that applications and data-processing systems which have been used in commercial but also in academic environments in the past cannot handle this new data volumes efficiently. Mostly, this regards the time and needed storage space for data processing and information retrieval. The research and development work in this area is generally summarized with the term *big data*. The main challenges in this field are search strategies, the distribution of information and the automated analysis of huge data volumes from a multitude of different, heterogeneous sources. In addition, the post-processing and visualization of collected data or extracted information is dealt with as well [3]. Considering the origin of *big data*, big data environments are mostly dependent on vast, centralized computing resources, such as high performance computing and high-speed storage systems. Typical use cases are data mining applications, scientific data evaluation and financial computing, such as fraud detection or customer pattern detection in the field of insurances [2]. When classifying applications or techniques as targeted for *big data* the *3Vs of big data* have been coined in [4]. There, the key elements of big data are defined as data *volume*, *velocity* with which new data is generated, and *variety* of the data items themselves.

In traditional data processing systems, relational databases are the data storage of choice whereas these do not suffice in big data applications, since they are neither capable to handle the ever increasing data volumes nor the rapid dataset changes and invalidations. Therefore, a transition to different technologies has happened in the field of big data environments. New systems such as NoSQL databases and Apache Hadoop have been created with massively distributed computing possibilities and distributed storage in mind.

2 Big data and Wireless Sensor Networks

In the past, Wireless Sensor Networks (WSN) focused on mostly homogeneous and rather resource-limited nodes which only acted as dull sensors delivering

their measurement data to a single node (the *sink*). The application logic would mainly sit on the sink and information retrieval (i.e. data processing) would happen there. As long as the number of nodes and the size of the data to transmit was rather low, this scenario worked usually well. Nonetheless, one main problem of such traditional sensor networks is, that they have been tailored for a specific application. Consequently, they can only react poorly to changes in application requirements or use cases, if at all. This often means that a change in the user's wishes requires to reflash or reconfigure every single node in the network. With the emergence of new use cases for wireless sensor/actuator networks in recent years, such as Car2X/Vehicle2X communication in the automotive context or Internet of Things (IoT) applications, and their massive information and data potential and nodes which are not under the user's control, new strategies are needed to handle these problems. For this evolving class of WSN applications we propose the term *big data* as well. At first sight, big data applications and WSN do not seem to have much in common, but we can rather easily show that key challenges in these two fields match, using the aforementioned 3V model as follows:

- **Volume:** A high node density or nodes which have the ability to deliver much different pieces of information (complexity)
- **Velocity:** The speed with which information changes in WSN (for instance individual sensor values)
- **Variety:** Networks with nodes which largely vary in hardware and software capabilities

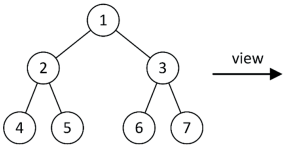
Therefore, by speaking of big data applications in WSN we focus on wireless, embedded sensor networks with data volumes larger than in traditional sensor network contexts [1]. Additionally, networks possessing a high density or just a vast absolute amount of nodes are considered. Like traditional big data applications, big data use cases in WSN incorporate networks with such a high node amount or complexity that traditional methods (e.g. duty cycling, forward-and-forget) cannot be used efficiently anymore. This comprises latency of the data delivery, storage requirements on the node level and energy consumption regarding the number of transmissions between nodes as well as the traffic volume. Due to the fact that hardware resources such as processing power, computing time and energy are strictly limited in WSN, the objective of big data applications in this field cannot be collecting as much information as possible [1]. Quite the contrary, the gathering of as little data as needed by still satisfying the user's need for information is required. Therefore, the main focus of research work does not lie on data mining in vast, existing database but on the decision which readily available data or information from a node has to be sent in which form to the sink. Naturally, in the automotive context (e.g. Car2X), energy resources as well as buffer storages are not as limited as in other mobile embedded sensor platforms.

Nevertheless, data processing and aggregation approaches have to use the given resources efficiently. Main concerns here, are latency, communication time and communication range. Another issue which needs to be dealt with is the high

mobility of nodes [5]. Last but not least, vehicles acting as sensor nodes provide us with more capabilities regarding sensor values and ultimately information which needs to be processed.

3 The approach

The approach on which we are focusing our efforts mainly incorporates a different view on WSN, where traditional data aggregation strategies reach their limits with regards to energy-efficiency and latency when facing big data applications. While traditional data processing systems moved away from relational databases, we propose this very concept for big data sensor networks, moving away from currently used proprietary, single application tailored WSNs. By establishing a database-oriented view on a Wireless Sensor Network we free the user from defining the actual use case before the network node distribution and can react to changing application requirements. The key here is the use of queries (like in traditional relational database systems) with which the user can define what kind of information they want to collect from the database. The queries are formulated against a *virtual table* which consists of all or some nodes of the sensor network represented as rows and where properties or actual sensors are modeled as columns, see Figure 1. Heterogeneous nodes can easily be represented by adding columns for which the value of nodes which do not possess this sensor is null. Each query then stands for a different application / use case of the sensor nodes, e.g. monitoring room temperature or activity. By allowing



Node ID	Parent ID	Temperature	Pressure
1	-	20.0 °C	100.0 kPa
2	1	22.0 °C	100.5 kPa
3	1	22.0 °C	100.0 kPa
4	2	21.5 °C	99.7 kPa
…	…	…	…

Fig. 1: Database-oriented view on a Wireless Sensor Network

the user to state queries, routing can be optimized in the network, by avoiding communication with nodes which are not needed to answer the query based on conditions as well as fuse sensor data at the node level [6]. A sample query is shown in Figure 2. If for example, the user wants to receive the maximum temperature value measured in a given node set, only the local maximum value has to be transmitted from node to node. In contrast, in traditional networks each node would send its individual value to the sink where the maximum would then be determined. Thus when using the database-oriented aggregation, communication traffic is effectively reduced. This enables a high potential for energy conservation in application-dependent WSNs and greatly supports the usage of

Wake-Up Receiver technology, which allows nodes which are idle and not required in a currently running query to shutdown their main transceiver module (e.g. WLAN, Bluetooth, Zigbee) and CPU and still be reachable. Changing application requirements as well as as different use cases which coexist at the same time can now easily be modeled as queries and a reflash or redistribution of sensor nodes is not necessary. The tests we carried out, based both on simulated

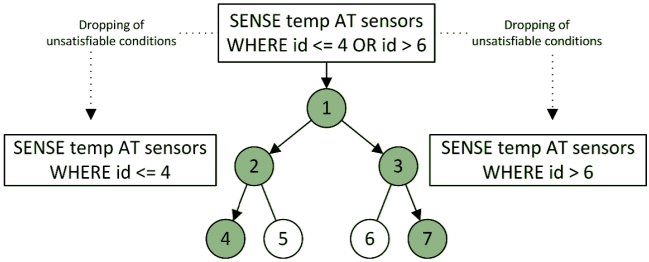


Fig. 2: A sample query in a database-oriented WSN which is only transmitted to nodes which satisfy the conditions

and on real hardware nodes, have confirmed the enormous potential regarding energy-efficiency in big data WSN and encourage us to continue research in this field.

References

1. Bergelt, R., Vodel, M., Hardt, W.: Energy efficient handling of big data in embedded, wireless sensor networks. In: Sensors Applications Symposium (SAS), 2014 IEEE. pp. 53–58 (Feb 2014)
2. Demchenko, Y., Zhiming, Z., Grosso, P., Wibisono, A., de Laat, C.: Addressing big data challenges for scientific data infrastructure. In: Proceedings of the IEEE 4th International Conference on Cloud Computing Technology and Science. pp. 614–617. IEEE Computer Society (2012)
3. Jung, G., Gnanasambandam, N., Mukherjee, T.: Synchronous parallel processing of big-data analytics services to optimize performance in federated clouds. In: Proceedings of the IEEE 5th International Conference on Cloud Computing. pp. 811–818. IEEE Computer Society (2012)
4. Laney, D.: The Importance of 'Big Data': A Definition. Gartner (2012)
5. Vodel, M., W. Hardt: Data aggregation and data fusion techniques in wsn/sanet topologies - a critical discussion. In: Proceedings of the TENCON 2012. pp. 1–6. IEEE Computer Society (November 2012)
6. Wang, L., Chen, L., Papadias, D.: Managing and Mining Sensor Data, chap. Query Processing in Wireless Sensor Networks, pp. 51–76. Springer US, Boston, MA (2013), http://dx.doi.org/10.1007/978-1-4614-6309-2_3
7. Wu, X., Zhu, X., Wu, G., Ding, W.: Data mining with big data. Transactions on Knowledge and Data Engineering 99 (June 2013)

Incremental DOM for Web Components

Markus Ast

Technische Universität Chemnitz, Germany
markus.ast@informatik.tu-chemnitz.de

Abstract The emerging standard of Web Components eases development and composition of client-side functionality of web applications. One- and two-way data bindings are a common and recurring problem for such applications. Providing means for easily integrating data bindings into Web Components decreases repetition and increases development efficiency. This work therefore proposes a way of utilizing virtual DOMs for integrating data binding into Web Components.

1 Introduction

Todays modern web applications are not re-rendered on the server on each user interaction. They are rather updated on the client-side in response to user interactions (or also in response to push updates to the server). Most of the time, after initially loading the website from the server, the only required subsequent communication with the server is to fetch data, e.g. through REST services. That is, handling updates on the client-side and updating the interface accordingly is a field that yielded many approaches in the last couple of years. Especially since these updates are often not just one-way, but also two-way, i.e., data changes lead to interface updates and interactions with UI elements can lead to changes to their underlying data.

For a good user experience, it is recommended to achieve 60 frames per seconds (fps) [3], i.e., updates of the Document Object Model (DOM) should not lead to lags when scrolling or when having animations in the web applications [2]. To consolidate expensive DOM operations Facebook's *React* populated the so called *Virtual DOM* approach, which is their main building block for achieving 60fps web applications. *React* creates the whole application through components to achieve separation of concerns. However, it is not based around the emerging Web Components standard [5].

This paper will therefore introduce an approach for using virtual DOMs in the standardized Web Components to allow for easily providing one- and two-way data bindings and therefore to ease development of such components. The work is structured as follows. At first, some background about virtual and incremental DOM in section 2, and about immutable data structures in Section 3 are given. Afterwards, the approach is introduced in Section 4. Finally, a conclusion is given in Section 5.

2 Virtual and Incremental DOM

The novel *Virtual DOM* approach has been populated by Facebook's *React*[1]. It arose from the goal of achieving 60 fps even on websites where data and therefore the DOM changes rapidly. DOM changes are expensive operations, because they often—depending on how well the website is made—lead to complete layout re-calculations. *Virtual DOM* is, in the end, about consolidating multiple DOM changes and applying them as one batch.

This it achieved by abstracting the DOM into a virtual DOM (VDOM). Meaning that the DOM is created through lightweight JavaScript objects. This lightweight virtual representation of the DOM is then used to create the actual DOM at once. On each subsequent change of the underlying data, the virtual DOM is re-created completely. This result is then compared to the previous virtual DOM. The resulting differences are used to patch the actual DOM in one operation. While re-creating the virtual DOM, comparing it with the previous one to extract differences and patch the actual DOM afterwards sounds like a whole bunch of work, it is—surprisingly—still faster than updating the actual DOM directly on each change of the underlying data. Especially when the amount of displayed data increases [1]. This procedure is shown in Figure 1 (a).

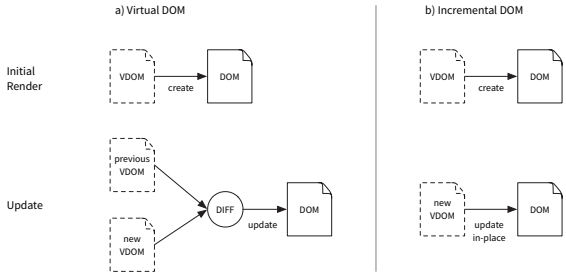


Figure 1. Virtual DOM (a) and Incremental DOM (b) render procedures

Google's *Incremental DOM*[2] (IDOM) is a variation of Facebook's *Virtual DOM*. It skips one step of the *Virtual DOM* procedure: it compares a virtual DOM with the actual DOM and updates it in-place directly. Rather than comparing two virtual DOMs and updating the actual DOM with patches created from the differences between both virtual DOMs. This reduces memory usage and further improves the overall performance [4]. This procedure is shown in Figure 1 (b). Both approaches can also be used to pre-render a view on the initial request to the server by creating a string representation of the resulting VDOM instead of creating an actual DOM.

[1] https://facebook.github.io/react/
[2] https://github.com/google/incremental-dom

3 Immutable Data

One major part in utilizing some kind of virtual DOM approach is to detect data changes to trigger the DOM updates accordingly. Doing so on each frame without having actual data changes would be an unreasonable amount of waste of CPU cycles, could reduce the overall performance and would have a negative impact on the battery life of mobile devices. To get immediate view updates on data changes, data has to be checked for updates as often as possible, e.g. even on each of the targeted 60 fps, or at least at 30 fps. This check must therefore be as little work as possible. Comparing huge nested data structures with each other would not be a good solution.

React therefore also populated the usage of immutable data in JavaScript. Immutable data structures cannot—once created—be changed. Subsequent changes always create a shallow copy in which the actual changes are integrated. While this requires some additional work when changing data, there are plenty of possibilities to optimize these writes; e.g. as done in Facebook's *immutable.js*[3].

However, this additional cost on data changes is low compared to the cost savings when comparing data for changes. Since each change creates a new data structure, it is sufficient to compare the pointers of two data structures in the first step. Data structures with the same pointer are identical. Having two distinct pointers indicates changes between two data structures. The actual changes are then identified by comparing the contents of both data structures with each other. For each nested data structure, the same procedure applies: first the pointer is compared and only if it is not equal, contents are compared. Having such a low cost update detection enables to check for updates even in browsers animation frames using the *window.requestAnimationFrame()* API.

4 Approach

Considering the structure of a Web Component: A Web Component uses HTML *template* tags to define and instantiate DOM content that is used for its own markup. The result is, according to the specification, created into a Shadow DOM [6]. That is, the starting point is not a virtual tree but an actual DOM. This work therefore proposes the following procedure to apply the virtual DOM approach to Web Components. Template markup, like variable placeholders, are integrated into the HTML directly. Control flow statements are provided through special template tags (which could be considered as own Web Components). An example is shown in Listing 1.1.

```
<template repeat="{{ todos as todo }}">
  <input type="radio" checked="{{ todo.isDone }}" />
  {{ todo.task }}
</template>
```

Listing 1.1. Example template markup statements

[3] https://facebook.github.io/immutable-js/

When the Web Component template is first loaded it is traversed completely to find and parse template tags but to also virtualize it. The result is a VDOM Factory, a JavaScript function that can be called with actual data and returns a virtual DOM. Using Google's *Incremental DOM*, this virtual DOM is used to update the actual DOM in-place. This is done after the Web Component is created but also on all subsequent data updates. This procedure is shown in Figure 2.

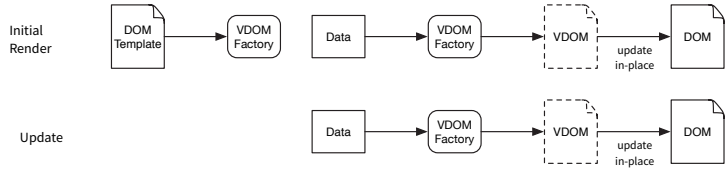


Figure 2. Incremental DOM from Web Component template render procedures

5 Conclusion

This work introduced a way of easily utilizing the approach of virtual DOM for Web Components. It thereby uses Google's Incremental DOM and recommends the usage of immutable data. The integration of this proposed approach is easily integrated into Web Components and provides one- and two-way data bindings. It therefore eases the development of such Web Components. Since it is not opinionated about what specific immutable data structure are to be used or about how other logic is implemented, it acts as a general purpose view layer. An examplary implementation can be found at `https://github.com/rkusa/ipoxy`.

References

1. Hunt, P.: The Secrets of React's Virtual DOM (FutureJS 2014) (2014)
2. Kinlan, P.: What do people want from a news experience? (Dec 2014), https://paul.kinlan.me/what-news-readers-want/
3. Lewis, P.: Rendering performance (2015), https://developers.google.com/web/fundamentals/performance/rendering/?hl=en
4. Ubl, M.: Introducing Incremental DOM (Jun 2015), https://medium.com/google-developers/introducing-incremental-dom-e98f79ce2c5f
5. W3C Working Group: Introduction to Web Components (Jun 2014), https://www.w3.org/TR/components-intro/
6. W3C Working Group: Shadow DOM (Dec 2015), https://www.w3.org/TR/shadow-dom/

Optimierung der medizinischen Dokumentation durch den Einsatz assistierender spracherkennender Systeme – Konzept für ein Promotionsvorhaben

Christina Lohr

christina.lohr@informatik.tu-chemnitz.de

1 Einführung

Klinische Einrichtungen sind in Deutschland durch das „Sozialgesetzbuch (SGB) Fünftes Buch (V) – Gesetzliche Krankenversicherung“ [soz89] verpflichtet, Krankheiten und Behandlungen vor allem für Abrechnungen zu dokumentieren. Die medizinische Dokumentation ist für Ärzte sehr zeitintensiv und nimmt täglich je nach Fachbereich und Einrichtung mehrere Stunden in Anspruch [BM03]. Umfangreiche Dokumente, wie Berichte komplizierter Operationen, werden diktiert und durch Schreibpersonal erfasst. Es kann vorkommen, dass ungeschriebene Berichte tagelang unbearbeitet liegen bleiben. Automatische Spracherkennung kann diesen Prozess optimieren, wenn die Systeme gegenüber der Erkennung robust genug sind. Viele klinische Einrichtungen nutzen derzeit digitale Diktiersysteme (z.B. von Philips), setzen jedoch aufgrund hoher Lizenzkosten und Fehlerquoten mit medizinischen Fachausdrücken sowie damit zusätzlichem Korrekturaufwand Spracherkennungssysteme größtenteils nicht ein. Aus erkannter Sprache können Informationen extrahiert werden. Durch Methoden der Informationsextraktion können Phrasen, wie Medikationen, direkt aus freien Texten erkannt und in eine Schnittstelle eines Krankenhausinformationssystems überführt werden. Daraus ergibt sich die Forschungsfrage, ob mit Spracherkennung, die aus archiverten Daten entwickelt wird und zusammen mit Informationsextraktion agiert, der Prozess der medizinischen Dokumentation einer Einrichtung optimiert werden kann. In den folgenden Abschnitten wird das Konzept eines Promotionsvorhabens vorgestellt, das diese Frage untersucht. Das Vorhaben wird in Kooperation mit dem Klinikum Chemnitz und weiteren klinischen Einrichtungen durchgeführt, wodurch für die Entwicklung von Komponenten Daten bereit gestellt werden.

2 Aktueller Stand und theoretische Aspekte der Sprachdatenverarbeitung in der Medizin

Für die Verarbeitung natürlicher Sprache existieren für verschiedene Aufgaben unterschiedliche Theorien und Umsetzungen. Medizinische Texte beinhalten viele Abkürzungen, Zahlen, Datumsangaben, Begriffe zusammen mit Ziffern oder in verschiedenen Schreibweisen, die auf englische und lateinische Terminologien zurückzuführen sind, sowie viele

orthografische Fehler, die aufgrund Zeitmangel nicht korrgiert werden können. Medizinische Dokumentationstexte müssen für eine Sekundärnutzung besonders aufbereitet werden, wenn aus archivierten Befundtexten Komponenten von Spracherkennungs- und Informationsextraktionssystemen modelliert werden [SLG15]. Spracherkennungssysteme zur Erkennung umfangreicher Vokabulare, bestehen aus einem Sprachmodell, das geschriebene Sprache betrachtet und durch N-Gramm-Statistiken über Wahrscheinlichkeiten von Wortgruppen eines vorhandenen Textkorpus repräsentiert wird, aus einem Akustikmodell, das in der Regel durch Hidden-Markov-Modelle aus bestehenden Akustikaufnahmen konstruiert wird, sowie aus einem Wörterbuch, das alle zu erkennenden Begriffe neben der üblichen Schreibweise auch in Form der Lautschrift enthält, [PK08] und [JM09]. Für Open-Source-Spracherkennungssysteme wie „CMU Sphinx“ existieren in deutscher Sprache verschiedene Korpora, z.B. aus dem Projekt „Voxforge“. Jedoch sind diese Korpora zu klein, enthalten nicht die notwendigen Fachbegriffe und können in deutschen Kliniken nicht verwendet werden. Das kommerzielle System „Dragon Medical“ wird aufgrund hoher Fehleranfälligkeit auch mit zusätzlichen Fachvokabularen oft nicht eingesetzt. Das speziell für die Pathologie angepasste System „Phatos“ wird hinsichtlich geringer Fehleranfälligkeit häufiger genutzt. Informationsextraktionssysteme ermitteln domänenspezifische Phrasen freier Texte und ignorieren dabei irrelevante Informationen. Durch definierte Regeln, die durch Lernverfahren oft in der Verwendung mit wissensrepräsentierenden Datenbanken trainiert werden, können relevante Abschnitte erkannt werden [Neu01]. Conditional Random Fields werden derzeit oft genutzt, um Informationsextraktionssysteme zu trainieren [LMP01]. Für die Segmentierung von Sequenzen werden ungerichtete grafische Modelle aufgebaut und beispielsweise durch das Gradientenverfahren traniert. Es ist möglich an jeder Stelle auf die gesamte Information der Eingabesequenz zuzugreifen. Das „Unified Medical Language System“ (UMLS) beinhaltet verschiedene Klassifikationssysteme und Thesauren aus dem medizinischen Umfeld, neben Englisch auch Deutsch. Durch die „CLEF eHealth Evaluation Challenge“ werden Forschungen für die medizinische Dokumentation, die Spracherkennung sowie Informationsextraktion nutzt, massiv voran getrieben. Im Rahmen der Bachelorarbeit [Loh12] wurde bereits gezeigt, dass es möglich ist, aus archivierten Operationsberichten Sprachmodelle sowie aus einigen weinigen akustischen Aufnahmen kleine Akustikmodelle und somit die wichtigsten Komponenten für ein Spracherkennungssystem aus eigenen Daten zu konstruieren. Die Masterarbeit [Loh15] zeigte, dass sich Sprachmodelle gezielt in Abhängkeit des Diagnoseschlüssels „ICD-10-GM“ und des „Operationen- und Prozedurenschlüssels“ konstruieren lassen, sodass bestimmte Wortschätze dadurch besser erkannt werden können.

3 Konzept

3.1 Analyse der Arbeitsabläufe im Klinikum Chemnitz und weiteren klinischen Einrichtungen

Zu Beginn werden Wartezeiten, temporale Überlasungen sowie verwendete Ressourcen von konkreten Abläufen der medizinischen Dokumentation der Kooperationspartner und die daraus resultierende Effizienz analysiert. Interviews mit Akteueren, die an den Arbeitsschritten beteiligt sind, und technische Schnittstellen, die zur möglichen Automatisierung

während der Ausführung in den Arbeitsabläufen verwendet werden, sollen untersucht werden. Das Ergebnis ist die Erschließung von Arbeitsschritten der medizinischen Dokumentation der Kooperationspartner hinsichtlich der Punkte für das Promotionsvorhaben als auch eine Kalkulation des Arbeitsaufwandes bezüglich der zeitlichen Last.

3.2 Korpusentwicklung für Spracherkennung

Die Optimierung von Spracherkennung hinsichtlich des Sprachumfangs im medizinischen Umfeld erfordert die Anwendung kontextspezifischer Sprachmodelle. Dafür werden textuelle Daten von anonymisierten Berichten akquiriert und so aufbereitet, dass sie für das Sprachmodelltraining nutzbar sind. Zur Evaluation der Spracherkennung wird ein Testdatensatz mit Audio-Mitschnitten und Transkripten als Referenz aufgebaut. Die Aufnahmen entsprechen dem typischen Szenario einer verbalen klinischen Dokumentation.

3.3 Implementierung von Spracherkennung

Das Ziel der Implementierung der Spracherkennung besteht darin, ein Spracherkennungssystem zur Transkription für verbale klinische Dokumentation zu entwickeln und anschließend zu evaluieren. Als technische Grundlage existieren bereits Open-Source-Werkzeuge, wie SRILM sowie CMU Sphinx und HTK. Um viele verschiedene Kontexte im klinischen Umfeld abdecken zu können, liegt der Fokus der Sprachmodellierung darin, Texte für Modelle hinsichtlich des Operationen- und Prozedurenschlüssels (OPS) [Ber15b] und der Klassifikation für Krankheiten und Diagnosen ICD-10-GM [Ber15a] zusammenzustellen. Für die akustische Modellierung sowie die Erstellung von Aussprachewörterbüchern wird auf vorhandene Datensammlungen der Professur Medieninformatik zurückgegriffen.

3.4 Korpusentwicklung für Informationsextraktion

Die Entwicklung maschineller Verfahren der Informationsextraktion setzt die Aufbereitung und intellektuelle Annotation großer Datenbestände voraus. Textuelle Daten verschiedener Berichte werden dafür zusammengetragen und inhaltlich untersucht. Anschließend werden Passagen zur thematischen Zuordnung wie Medikationen, Krankheitsbilder oder konkrete Inhalte annotiert. Dabei werden vorhandene Klassifikationssystememe genutzt, die durch das „Unified Medical Language System“ (UMLS) zusammengefasst sind.

3.5 Implementierung von Informationsextraktion

Inhaltlich relevante Bereiche des verbalen Dokumentationstextes, der für die Informationsextraktion aufbereitetet wird, werden identifiziert und semantisch aufbereitet. Es ist dadurch möglich, Informationen hinsichtlich Patienten, Diagnosen, Behandlungen, Medikationen sowie weiteres Vorgehen zu ermitteln und zeitnah in bestehende Systeme zu überführen. Für die Konstruktion eines entsprechenden Modells und Evaluation wird der Korpus verwendet, der bereits im vorgehenden Abschnitt beschrieben wurde. Das Ziel besteht darin, ein Verfahren zur automatischen Informationsextraktion in verschriftlichten klinischen Texten zu entwickeln, um vereinzelte Attribute in Formular zu überführen.

3.6 Evaluation optimierter Arbeitsabläufe

Anhand der entwickelten und speziell für den Einsatz in der Medizin angepassten Spracherkennungs- und Informationsextraktionssysteme wird der Aufwand in den Arbeitsabläufen der medizinischen Dokumentation der Kooperationspartner evaluiert. Die beiden Verfahren werden unabhängig und kombiniert betrachtet, wobei die Schwelle für den positiven Gewinn an Effizienz untersucht wird. Das Ergebnis besteht aus der Überprüfung der Eignung der im Vorhaben verwendeten Methoden für das Dokumentationswesen in klinischen Einrichtungen.

Literatur

[Ber15a] Bernd Graubner. *ICD-10-GM 2016 Systematisches Verzeichnis: Internationale statistische Klassifikation der Krankheiten und verwandter Gesundheitsprobleme*. Deutscher Ärzte-Verlag, Köln, 1. Auflage, 2015.

[Ber15b] Bernd Graubner. *OPS 2016 Systematisches Verzeichnis: Operationen - und Prozedurenschlüssel Internationale Klassifikation der Prozeduren in der Medizin*. Deutscher Ärzte-Verlag, Köln, 1. Auflage, 2015.

[BM03] Karl Blum und Udo Müller. Dokumentationsaufwand im ärztlichen Dienst der Krankenhäuser. *das Krankenhaus*, 95(7):544–548, 2003.

[JM09] Jurafsky, Daniel und Martin, James H. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. Pearson Education Internatinal, Second Edition, 2009.

[LMP01] John Lafferty, Andrew McCallum und Fernando CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data, 2001.

[Loh12] Christina Lohr. Sprachmodelladaption von CMU Sphinx für den Einsatz in der Medizin und dessen Evaluierung. Bachelerarbeit, TU Chemnitz, 03 2012.

[Loh15] Christina Lohr. Entwicklung und Evaluierung von Sprachmodellen auf Basis des Operationen- und Prozedurschlüssels für den Einsatz von Spracherkennungssystemen in der medizinischen Dokumentation. Masterarbeit, TU Chemnitz, 08 2015.

[Neu01] Günter Neumann. Informationsextraktion, 2001.

[PK08] Beat Pfister und Tobias Kaufmann. *Sprachverarbeitung: Grundlagen und Methoden der Sprachsynthese und Spracherkennung*. Springer, Berlin; Heidelberg, 2008.

[SLG15] Stefan Schulz und Pablo López-García. Big Data, medizinische Sprache und biomedizinische Ordnungssysteme. *Bundesgesundheitsblatt-Gesundheitsforschung-Gesundheitsschutz*, Seiten 1–9, 2015.

[soz89] Das Fünfte Buch Sozialgesetzbuch – Gesetzliche Krankenversicherung – (Artikel 1 des Gesetzes vom 20. Dezember 1988, BGBl. I S. 2477, 2482), das durch Artikel 2 des Gesetzes vom 15. April 2015 (BGBl. I S. 583) geändert worden ist, Januar 1989.

Attentive Robot Vision

Amirhossein Jamalian, Frederik Beuth and Fred H. Hamker

Technische Universitat Chemnitz, Artificial Intelligence
amirhossein.jamalian@informatik.tu-chemnitz.de
frederik.beuth@informatik.tu-chemnitz.de
fred.hamker@ informatik.tu-chemnitz.de

Visual attention is a smart mechanism to omit the processing of unnecessary data coming into primate's visual system. Based on this mechanism, primates can rapidly find and focus on the searched object in their visual field in presence of several dictractors and in front of a cluttered background. These find and focus processes are performed in the primates' brain through *ventral* and *dorsal streams* depicted in Figure 1 a. The former is involved in the object recognition task whereas the latter is the responsible of saccade generation and saliency map formation (Beuth & Hamker, 2015a).

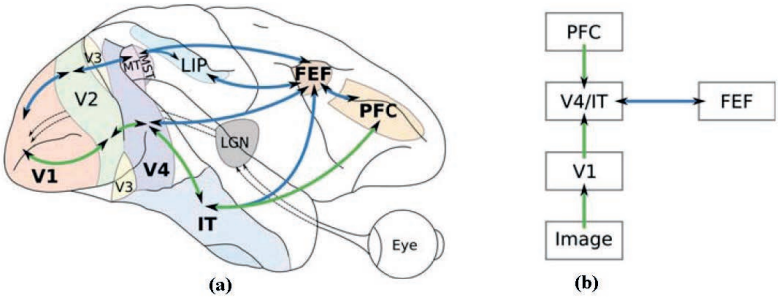


Figure 1. a) Primates' Visual System with ventral and dorsal streams, green and blue arrows respectively. b) Areas and connections which are simulated in the computational model. V1, V4, IT, PFC and FEF are primary visual cortex, fourth visual cortex, inferior temporal cortex, prefrontal cortex and frontal eye field respectively.

Heretofore, the biologically plausible computational model of visual attention, which is an implementation of the ventral stream, has been developed based on primates' ventral stream (Beuth & Hamker, 2015b). Inspired by cortical architecture, this model uses a structure with two layers to implement amplification, normalization and suppression as well as spatial pooling. The diagram of its layers and corresponding connections is illustrated in Figure 2. The input image immediately is preprocessed (as the effect of the Lateral Geniculate Nucleus (LGN)) and stays in the neurons of V1 layer. Then, it is given to the cycle of HVA-FEF to recognize and localize the searched object concurrently (HVA is abbreviation of Higher Visual Area whereas FEF is abbreviation

of Frontal Eye Field). Therefore, it could be categorized in the class of iterative models mentioned in (Jamalian & Hamker, 2016). HVA is the combination of V2, V3, V4 and IT regions in the brain. The performance of current model in Object Recognition of COIL-100 dataset (Nene, Nayar, & Murase, 1996) is 92%, 72% and 42% in black, white-noise affected and real-world backgrounds respectively (Beuth & Hamker, 2015a).

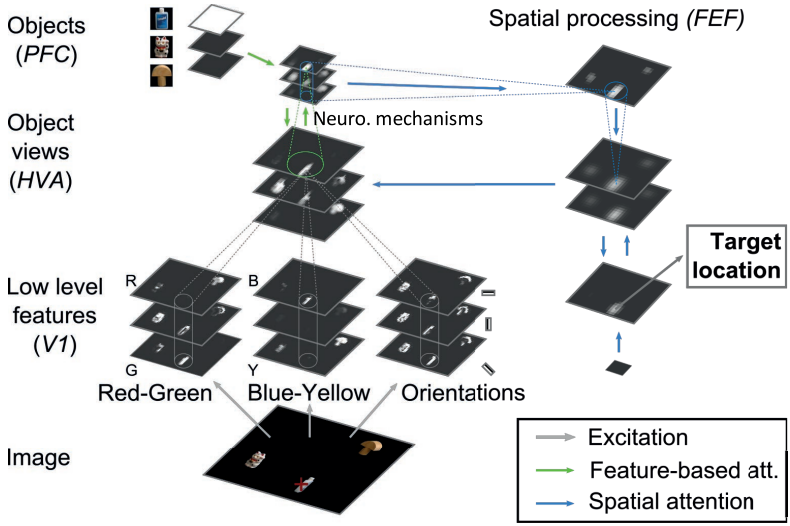


Figure 2. The computational model of visual attention in search of a bottle in COIL-100 dataset (Nene, Nayar, & Murase, 1996). The picture is adapted from (Beuth & Hamker, 2015a).

Recently, this model, individually as well as integrated with the model of dorsal stream, has been tested using a neuro-cognitive agent (Hamker, 2015) in virtual reality (VR) within the European Union project so-called *Spatial Cognition*. This is the first time that the robustness of such models against large scaling for the visual search task (Wolfe, 1994) in semi real-world scenarios is demonstrated. Few models have already been demonstrated with (semi) real-world objects (Chikkerur, Serre, Tan, & Poggio, 2010) & (Walther & Koch, 2007). Yet, they have mostly used static input material.

The VR is based on the VR engine Unity[1], while the neuro-cognitive agent is implemented using ANNarchy which is a new neural network simulator reported in (Vitay, Dinkelbach, & Hamker, 2015). We designed a set of scenarios to test the capability of the agent in object recognition and localization. In each single scenario, the agent has to execute a classical visual search task (Wolfe, 1994), i.e. search a target object among distractors. For instance, Figure 3 a. demonstrates the scene in which the agent should search for a yellow crane truck (target) while all other objects serve as

[1] https://unity3d.com/

distractors. The set of scenarios contains 9000 different tasks; each is a visual field with at least 3 objects (one target and a couple of distractors). The performance of the model at this test set is depicted in Figure 3 b as a confusion matrix. As it can be seen in this figure, in 85% of the cases the model can localize the target correctly (the saccade landed within the object borders or not more than 50 pixels away).

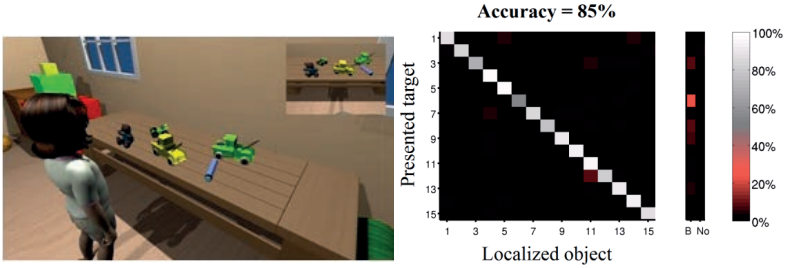


Figure 3. a) Neuro-cognitive agent in VR performing the visual search task. b) The performance of the object recognition / localization model based on 9000 scenarios.

In near future, first, we will implement the model on the iCub robot (a humanoid robot with the capability of saccade) to evaluate its performance in real-world scenarios. Then, we will improve the model based on the following ideas:

- Propagating the feature-based attention back to the V1 layer which is absent in current model. The idea is that, this attention signal could make the model more robust against noises and effect of distractors.
- Considering the robustness of the model against external and internal noises. It has been shown that the proper amount of noise can have positive effect in human visual attention through the mechanism called Stochastic Resonance (Kitajo, Yamanaka, Ward, & Yamamoto, 2006).
- Implementing the dynamic receptive field inside the model.

Acknowledgement

This work has been supported by the European Union's 7th Framework Program (FET, Neuro-Bio-Inspired Systems: Spatial Cognition) under grant agreement *n*° 600785.

References

Beuth, F., & Hamker, F. H. (2015b). A mechanistic cortical microcircuit of attention for amplification, normalization and suppression. *Vision Research* , 241-257.

Beuth, F., & Hamker, F. H. (2015a). Attention as cognitive, holistic control of the visual system. *Workshop of New Challenges in Neural Computation (NCNC 2015)*, (pp. 133-140). Aachen.

Chikkerur, S., Serre, T., Tan, C., & Poggio, T. (2010). What and where: a Bayesian inference theory of attention. *Vision Research , 50* (22), 2233-2247.

Hamker, F. H. (2015). Spatial Cognition of humans and brain-like artificial agents. *Künstliche Intelligenz* , 83-88.

Jamalian, A., & Hamker, F. H. (2016). Biologically-Inspired Models for Attentive Robot Vision: A Review. In R. Pal, *Innovative Research in Attention Modeling and Computer Vision Applications* (pp. 69-98). Hershey, PA: Information Science Reference.

Kitajo, K., Yamanaka, K., Ward, L. M., & Yamamoto, Y. (2006). Stochastic resonance in attention control. *Europhysics Letters* , 1029-1035.

Nene, S. A., Nayar, S. K., & Murase, H. (1996). *Columbia Object Image Library (COIL-100)*. Technical Report: CUCS-006-96.

Vitay, J., Dinkelbach, H. Ü., & Hamker, F. H. (2015). ANNarchy: a code generation approach to neural simulations on parallel hardware. *Frontiers in Neuroinformatics* , 1-20.

Walther, D. B., & Koch, C. (2007). Attention in hierarchical models of object recognition. *Progress in Brain Research , 165*, 57-78.

Wolfe, J. M. (1994). Guided search 2.0 a revised model of visual search. *Psychonomic Bulletin & Review* , 202-238.

Author Index

A

Ast, Markus	153	Incremental DOM for Web Components

B

Benndorf, Maik	99	Untersuchungen und Optimierung der Kontexterkennung auf mobilen Endgeräten mittels Sensorfusion
Bergelt, René	149	Big Data Applications in Wireless Sensor Networks
Beuth, Frederik	87	Entwicklung einer virtuellen Versuchsumgebung zur experimentellen Untersuchung von Raumorientierung und visueller Aufmerksamkeit
	111	Reinforcement Learning with Object Localization in a Virtual Environment
	127	Learning of Spatial Invariances for Object-Ground Separation
	161	Attentive Robot Vision
Blokzyl, Stephan	145	Generisches Framework zur Parallelisierung von Echtzeitfähiger Bilddatenauswertung für Rekonfigurierbare Integrierte Schaltkreise
Burian, Frank	141	CloudLinkSync: URI-basierte Adressierung lokaler Kopien von Cloud-Ressourcen

D

Dinkelbach, Helge	109	Reinforcement Learning with Object Localization in a Virtual Environment

G

Gussev, Joseph	111	Reinforcement Learning with Object Localization in a Virtual Environment

H

Hamker, Fred	87	Entwicklung einer virtuellen Versuchsumgebung zur experimentellen Untersuchung von Raumorientierung und visueller Aufmerksamkeit
	111	Reinforcement Learning with Object Localization in a Virtual Environment
	127	Learning of Spatial Invariances for Object-Ground Separation
	161	Attentive Robot Vision

Hardt, Wolfram	105	Schallquellenlokalisierung zur Sprecheridentifizierung für Multi-User Dialogschnittstellen
	145	Generisches Framework zur Parallelisierung von Echtzeitfähiger Bilddatenauswertung für Rekonfigurierbare Integrierte Schaltkreise
	149	Big Data Applications in Wireless Sensor Networks

J

Jakobs, Christine	1	Holistische Beschreibung der Fähigkeiten heterogener Robotergruppen
Jakobs, Thomas	27	Nicht-blockierende Implementierung von verketteten Listen mit Hilfe der Intel Transactional Synchronization Extension
Jamalian, Amirhossein	161	Attentive Robot Vision
Jüngel, Sascha	87	Entwicklung einer virtuellen Versuchsumgebung zur experimentellen Untersuchung von Raumorientierung und visueller Aufmerksamkeit

K

Keller, Titus	75	Entwicklung eines webbasierten Curve-Fitting Tools für komplexe Multiparameter-Funktionen
Kendzia, Kevin	37	Konzeption und Implementierung einer Sprachsteuerung für Roboter
Kowerko, Danny	75	Entwicklung eines webbasierten Curve-Fitting Tools für komplexe Multiparameter-Funktionen
Krug, Michael	133	Distributed Event-based Communication for Web Components
Kuhnert, Hannes	117	Automatische Berechnung des Ruiz-Kriteriums im Finite-Elemente-Postprocessing

L

Lohr, Christina	157	Optimierung der medizinischen Dokumentation durch den Einsatz assistierender spracherkennender Systeme - Konzept für ein Promotionsvorhaben

M

Manthey, Robert	63	Entwurf und Implementierung von Optimierungs- und Funktionserweiterungen der hexagonalen Bildrasterung in der Videokompressionssoftware x264HMod

N

Naumann, Stefan	13	Automatisierte Verlässlichkeitsmodelle verteilter Anwendungen

R

Ritter, Marc	63	Entwurf und Implementierung von Optimierungs- und Funktionserweiterungen der hexagonalen Bildrasterung in der Videokompressionssoftware x264HMod
	75	Entwicklung eines webbasierten Curve-Fitting Tools für komplexe Multiparameter-Funktionen

S

Schlosser, Tobias	63	Entwurf und Implementierung von Optimierungs- und Funktionserweiterungen der hexagonalen Bildrasterung in der Videokompressionssoftware x264HMod
Schmidt, René	105	Schallquellenlokalisierung zur Sprecheridentifizierung für Multi-User Dialogschnittstellen
Schwarz, Alex	127	Learning of Spatial Invariances for Object-Ground Separation

W

Wiedemann, Fabian	137	COP: Improving Collaboration in e-Health Utilizing the Cloud

Z

Zywietz, Anna	51	Evaluation von Touch-Gesten zur Steuerung eines nicht-humanoiden Roboters