Modulhandbuch Informatik Master of Science

Version M_Inf25.1_S

Letzte Änderung: 2026-07-11 15:25:33

Inhaltsverzeichnis

Module

◈ MM003 – Algorithmics

Verantwortliche:	Sebastian Iwanowski
Moduldauer:	6 Monate
Unterrichtssprache:	english

Bestandteile:

Teilleistung:	TM027 – Algorithmics
Lernform:	Vorlesung
Prüfungsform:	Klausur / Mündliche Prüfung
Prüfungsdauer:	
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	30 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	60 Stunden
Aufwand während Semesterferien:	60 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	150 Stunden
Lehrende:	Sebastian Iwanowski

Studieninhalte:

- Einführung in die formale Algorithmik
  - Vergleich der grundlegenden Sortiertechniken
  - Komplexitätsmaße für die Analyse von Algorithmen
  - Untere Schranke für Algorithmen, die nur Vergleiche verwenden
- Fortgeschrittenes Suchen und Sortieren
  - Ordnungsstatistik
  - Suche in sortierten Arrays
  - Sortierung in endlichen Bereichen
- Lösungen für das Wörterbuchproblem
  - Hashing und andere Methoden zur Optimierung des durchschnittlichen Laufzeitverhaltens
  - (2,3)-Bäume als Beispiel für einen optimalen Baum für die schlechteste Laufzeit
  - Andere optimale Schlechteste-Fall-Methoden für Suchbäume
  - Optimale binäre Suchbäume (Bellman)
- Graphenalgorithmen
  - Die Erstellung minimal aufspannender Gerüste als Motivation für grundlegende Algorithmen
  - Kürzeste Wege (Dijkstra, Floyd-Warshall, Straßen)
  - Berechnung der maximalen Flüsse in q/s-Netzwerken (Ford-Fulkerson, Edmonds-Karp, Dinic)
  - Berechnung von Graphenmatchings (bipartit, Edmonds)
- String-Matching
- Grundlagen der algorithmischen Geometrie
  - Grundlegende Probleme und die Verwendung von Voronoi-Diagrammen zu ihrer Lösung
  - Sweep-Techniken (einschließlich Berechnung von Voronoi-Diagrammen)

Lernergebnisse:

Die Studierenden ...

- kennen die grundlegenden Problemstellungen der Algorithmik und deren klassische Lösungsverfahren.

- können die Korrektheit und Effizienz von Algorithmen analysieren.
- haben detaillierte Kenntnisse über fortgeschrittene Algorithmen für diverse Problemstellungen in ausgewählten Anwendungsbereichen.
- wissen, wie man theoretische Ergebnisse in praktischen Anwendungen implementiert.

Voraussetzungen und Empfehlungen:

Diskrete Mathematik aus dem Bachelorstudium, gutes allgemeines Programmierverständnis

Literatur:

- deBerg, M., Cheong, O., van Krefeld, M., Overmars, M.:
  Computational Geometry, Algorithms and Applications.
  Springer 2008 (3. edition), ISBN 978-3540779735
- Cormen, T.; Leiserson C.; Rivest, R.; Stein, C.:
  Introduction to Algorithms,
  MIT Press 2001 (2nd ed.)
- Levitin, A.:
  Introduction to the Design and Analysis of Algorithms.
  Addison-Wesley 2006, ISBN 0-321-36413-9
- Mehlhorn, K. / Sanders, P.:
  Algorithms and Data Structures The Basic Toolbox.
  Springer 2008, ISBN 978-3-540-77977-3
- Papadimitriou, C. / Steiglitz, K.:
  Combinatorial Optimization Algorithms and Complexity.
  Dover 1998, ISBN 0-486-40258-4

Studiengänge:

- Data Science & Artificial Intelligence Master of Science
- Informatik Master of Science
- IT Engineering Master of Science

◈ MM005 – Funktionale Programmierung

Verantwortliche:	Torben Tietgen
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM028 – Funktionale Programmierung
Lernform:	Vorlesung
Prüfungsform:	Klausur / Mündliche Prüfung
Prüfungsdauer:	45 Min.
ECTS:	2.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	2 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	15 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	25 Stunden
Aufwand während Semesterferien:	20 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	60 Stunden
Lehrende:	Frank Huch

Teilleistung:	TM029 – Übg. Funktionale Programmierung
Lernform:	Übung
Prüfungsform:	Abnahme
Prüfungsdauer:	10 Min.
ECTS:	3.0
Benotung:	Bestanden/nicht Bestanden
Turnus:	jährlich
Dauer (pro Termin):	2 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	15 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	75 Stunden
Aufwand während Semesterferien:	0 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	90 Stunden
Lehrende:	Torben Tietgen

Studieninhalte:

- Syntax von Haskell
- Produkt- und Summen-Datentypen
- Rekursive Datenstrukturen
- Polymorphismus
- Funktionen höherer Ordnung
- Hindley-Milner Typsystem, Typüberprüfung und Typinferenz
- Korrektheitsargumentationen
- Funktionale Datenstrukturen
- Bedarfsauswertung und unendliche Strukturen
- Typklassen
- Funktoren und Monaden
- Beispielanwendungen

Zusätzlich werden in der Übung praxisrelevante Aspekte der Anwendungsentwicklung mit der Programmiersprache Haskell behandelt, die nicht Bestandteil der Vorlesung sind. Die Bearbeitung der Übungsaufgaben folgt parallel zum Stoff der Vorlesung in Zweiergruppen.

Lernergebnisse:

Die Studierenden ...
- beherrschen fortgeschrittene Techniken der funktionalen Programmierung am Beispiel der Sprache Haskell.
- können mit Funktionen höherer Ordnung, mit polymorphen Datentypen und Typklassen, mit Funktoren, Monaden, Monoiden und weiteren mathematischen Strukturen umgehen und in neuen Kontexten anwenden.
- beherrschen Fähigkeiten in der Modellbildung und Abstraktion und können sie anwenden.
- kennen die Bezüge zwischen Mathematik und funktionaler Programmierung.
- kennen die Vor- und Nachteile des funktionalen Paradigmas für Anwendungen der IT-Sicherheit.

Voraussetzungen und Empfehlungen:

Voraussetzungen sind Kenntnisse und praktische Erfahrungen in höheren Programmiersprachen, insbesondere mit getypten Sprachen. Außerdem werden Kenntnisse über Diskrete Mathematik und algebraische Strukturen erwartet.

Elementares Wissen über Komplexitätstheorie wird ebenfalls vorausgesetzt.
Zudem ist die Installation von Haskell auf dem eigenen Rechner empfehlenswert.

Literatur:

• Miran Lipovaca: Learn You a Haskell for Great Good!: A Beginner's Guide, No Starch Press, 2011, ISBN: 978-1593272838
• Graham Hutton: Programming in Haskell, Cambridge University Press, 2016, ISBN: 978-1316626221
• Richard Bird: Introduction to Functional Programming using Haskell, 2nd Edition Prentice Hall, New Jersey, 1998, ISBN: 978-0134843469

Studiengänge:

- Informatik Master of Science
- IT-Sicherheit Master of Science

◈ MM009 – Workshop Kryptographie

Verantwortliche:	Gerd Beuster
Moduldauer:	6 Monate
Unterrichtssprache:	english

Bestandteile:

Teilleistung:	TM030 – Workshop Kryptographie
Lernform:	Workshop
Prüfungsform:	Abnahme
Prüfungsdauer:	15 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	30 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	120 Stunden
Aufwand während Semesterferien:	0 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	150 Stunden
Lehrende:	Gerd Beuster

Studieninhalte:

- Theorie der Kryptographie
  - Semantische Sicherheit
  - Unbrechbare Verschlüsselung und One Time Pad
  - Diffusion und Konfusion
- Klassische Kryptographie
  - Substitution und Transposition
  - Affine Verschlüsselung
  - Rotormaschinen
- Moderne Kryptographie
  - Stream- und Block-Chiffren
  - DES und GOST
  - AES
- Blockchiffrierung Betriebsarten
  - ECB, CBC, CTR, AES-GCM
- Zufallszahlengeneratoren
  - TRNG und PRNG
  - Voraussetzungen für CSPRNG
- Hashing
  - Hashing-Algorithmen
    - SHA 2
    - SHA 3
  - Nachrichten-Authentifizierung
    - CMAC und HMAC
- Asymmetrische Kryptographie
  - Diffie-Hellman
  - Elliptische Kurven
  - Asymmetrische Verschlüsselung und digitale Signaturen
- Post-Quantum Cryptography
  - CRYSTALS-Kyber
- Hardware-Kryptographie
  - Trusted Computing
  - Smartcards
  - Differential Power Analysis

Lernergebnisse:

Nach Abschluss des Moduls sind die Studierenden in der Lage, ...

- Sicherheitswerkzeuge als wesentlichen Baustein moderner Informations- und Kommunikationssysteme zu verwenden.
- ihr Wissen über alle relevanten Aspekte der Daten-, Netzwerk- und Websicherheit anzuwenden.
- die Anwendung kryptographischer Methoden, insbesondere zur Authentifizierung, Verschlüsselung und Integritätserhaltung zu betrachten.
- die algorithmischen Stärken und Schwächen der kryptographischen Verfahren zu bewerten.
- kryptographische Protokolle zu bewerten und zu implementieren, insbesondere für die Authentifizierung im E-Commerce.
- alle für die Implementierung und Anwendung kryptographischer Methoden relevanten Nebenbedingungen zu berücksichtigen.
- die Qualität von Zufallszahlengeneratoren zu beurteilen.
- die Eignung von Software- und Hardware-Kryptographie für eine bestimmte Aufgabe abzuschätzen.

Voraussetzungen und Empfehlungen:

Für dieses Modul sind Grundkenntnisse der diskreten Mathematik erforderlich. Weiterhin müssen die Studierenden über grundlegende Kenntnisse der Programmierung verfügen, insbesondere in der Programmiersprache Python.

Literatur:

- Bos, Joppe, et al. : CRYSTALS – Kyber: A CCA-secure module-lattice-based KEM. In: O'Conner, L. (ed.) : Proceedings of the 2018 IEEE European Symposium on Security and Privacy (EuroS&P). London, UK, 2018, pp. 353-367, doi: 10.1109/EuroSP.2018.00032.
- Ferguson, Niels; Schneier, Bruce; Kohno, Tadayoshi: Cryptography Engineering : Design Principles and Practical Applications. Indianapolis (IN), USA: Wiley Publishing, 2010.
- Katz, Jonathan; Lindell, Yehuda : Introduction to Modern Cryptography. Boca Raton (FL), USA: CRC Press, 2007.
- Mao, Wenbo: Modern Cryptography: Theory and Practice, Upper Saddle River (NJ), USA: Prentice Hall, 2003.
- NIST : SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. FiPS PUB 202, https://doi.org/10.6028/NIST.FIPS.202 , 2015.
- Ristic, Ivan : Bulletproof TLS and PKI. 2. Edition. London, UK: Feisty Duck, 2022.
- Stallings, William: Cryptography and Network Security : Principles and Practice. 8. Edition. London, UK: Pearson, 2022.
- Stinson, Douglas R. : Cryptography : Theory and Practice. 4. Edition. Boca Raton (FL), USA: CRC Press, 2018.
- Swenson, Christopher: Modern Cryptanalysis : Techniques for Advanced Code Breaking. Indianapolis (IN), USA: Wiley Publishing, 2008.

Studiengänge:

- Informatik Master of Science
- IT-Sicherheit Master of Science
- IT Engineering Master of Science

◈ MM018 – Robotics

Verantwortliche:	Ulrich Hoffmann
Moduldauer:	6 Monate
Unterrichtssprache:	english

Bestandteile:

Teilleistung:	TM032 – Robotics
Lernform:	Vorlesung mit integrierter Übung
Prüfungsform:	Portfolio-Prüfung
Prüfungsdauer:	60 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	30 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	108 Stunden
Aufwand während Semesterferien:	0 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	138 Stunden
Lehrende:	Ulrich Hoffmann

Studieninhalte:

- Aufbau und Zusammensetzung von Robotern
  - Kinematik
  - Bewegung und Beweger
  - Effektoren
  - Programmier-Systeme
- Bewegungsmodellierung
  - Punkt-zu-Punkt-Steuerung
  - Interpolation von Trajektorien
- Modellierung von Aktionen
- Intelligente Sensoren
  - Taktile Sensoren
  - Optical sensors
- Lernende Roboter
- Praktisches Projekt
  - Eigenverantwortliches Umsetzen eines Projektes innerhalb des komplexen Themengebiets
  - Experimentelles Erforschen neuer Ansätze und Ideen, welche über den Vorlesungsinhalt hinaus gehen
  - Regelmäßige Diskussion der Projektergebnisse und Präsentationen vor allen Gruppen

Lernergebnisse:

Studierende

- verfügen über Grundkenntnisse ausgewählter Konzepte und Technologien der Robotik.
- verstehen vor allem die Eigenschaften mobiler autonomer Systeme gründlich.
- haben ein tiefes Verständnis der technischen Grundlagen der Robotik und insbesondere der Konzepte der Bewegungs- und Aktionsmodellierung sowie intelligenter lernender Sensoren als Grundlage des autonomen Roboterverhaltens.
- sind in der Lage, exemplarische Implementierungen der vorgestellten theoretischen Konzepte in einem selbstorganisierten und gruppenorientierten Projekt zu realisieren.
- können ausgehend von den vorgestellten Konzepten selbstständig neue Lösungsansätze entwickeln, umsetzen und das Ergebnis beurteilen.
- haben die Kompetenz, praktische Probleme zu verstehen, die auftreten, wenn Roboteraktionen durch visuelle Bilder gesteuert werden.

- sind in der Lage, ihre wissenschaftlichen Ergebnisse in einer geeigneten Präsentation mit geeigneten Präsentationstechniken verständlich zu vermitteln.
- sind in der Lage, komplexe wissenschaftliche Sachverhalte in einem Fachgespräch kompetent zu vermitteln.

Voraussetzungen und Empfehlungen:

- Kenntnisse in Programmier-Systemen und der Entwicklung imperativer Porgramme
- Fähigkeit zur eigenverantwortlichen Umsetzung eines Projektes innerhalb des komplexen Themengebiets der Robotik.
- Kompetenzen im experimentellen Erforschen neuer Ansätze und Ideen, die über den Vorlesungsinhalt hinausgehen, sowie in der regelmäßigen Diskussion und Präsentation von Projektergebnissen.
- Kentnisse in Linearer Algebra
- Grundlegende Fähigkeiten imperative Programme zu erstellen und auf Software-Bibliiotheken zuzugreifen
- Grundkenntnisse und grundlegende Fähigkeiten in der Programmierung von Bildverarbeitungsalgorithmen und der Benutzung einschlägiger Bibliotheken.

Literatur:

- Blume, Dillmann: Frei Programmierbare Roboter, Vogel Verlag, 1981
- McKerrow: Introduction to Robotics: Introduction to Robotics, Addison Wesley, 1991
- Stienecker: The KUKA Robot Programming Language, Eigenverlag, 2011
- Vukobratovic: Introduction to Robotics, Springer, 1989

Studiengänge:

- Data Science & Artificial Intelligence Master of Science
- Informatik Master of Science
- IT Engineering Master of Science

◈ MM023 – Seminar Informatik

Verantwortliche:	Ulrich Hoffmann
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM024 – Seminar
Lernform:	Seminar
Prüfungsform:	Schriftl. Ausarbeitung (ggf. mit Präsentation)
Prüfungsdauer:	60 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	Sommersemester
Dauer (pro Termin):	1 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	7 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	135 Stunden
Aufwand während Semesterferien:	0 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	142 Stunden
Lehrende:	*Ulrich Hoffmann*

Studieninhalte:

Fachvorträge mit anschließender Gruppendiskussion.

Lernergebnisse:

Die Studierenden ...

- sind in der Lage, eine wissenschaftliche fundierte Lösung für theoretische und/oder praktische Problemstellungen primär aus dem Themengebiet sowie ähnlichen Gebieten zu entwickeln.
- zeigen eine verbesserte Problemlösungstechnik, sicherere Verwendung von Termini, präzise Strukturierung im Aufbau schriftlicher Arbeiten und Einhalten der Formalia.
- zeigen eine auf Masterniveau angemessene Vortragstechnik im Rahmen der Präsentation der Ergebnisse.

Voraussetzungen und Empfehlungen:

- Fähigkeit, wissenschaftliche Themen zu strukturieren und eine eigenständige Zielsetzung zu entwickeln.
- Kenntnisse im Recherchieren und Aufbereiten wissenschaftlicher Inhalte.
- Vertrautheit mit den formalen Anforderungen und Kriterien wissenschaftlicher Arbeiten.
- Fähigkeit, formale Richtlinien sicher zu beachten und anzuwenden.
- Fähigkeit, schriftliche Ausarbeitungen größeren Umfangs zu erstellen.
- Kompetenzen im Vortragen und Diskutieren der Ergebnisse mit anderen Seminarteilnehmern.

Literatur:

Recherche nach aufgabenbezogener Literatur, teilweise aufgabenspezifische Vorgabe einzelner Literaturquellen.

Studiengänge:

- Informatik Master of Science

◈ MM062 – Praktikum Virtuelle Realität und Simulation

Verantwortliche:	Christian-Arved Bohn
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM038 – Prakt. Virtuelle Realität und Simulation
Lernform:	Projektarbeit
Prüfungsform:	Abnahme
Prüfungsdauer:	20 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	Sommersemester
Dauer (pro Termin):	5 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	37 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	48 Stunden
Aufwand während Semesterferien:	20 Stunden
Flexibel einteilbarer Aufwand:	64 Stunden
Gesamtaufwand:	*Gesamtaufwand 169 Stunden weicht signifikant von erwarteten 5.0 ECTS ab.*
Lehrende:	Christian-Arved Bohn

Studieninhalte:

Programmierung in C. Einarbeitung in das Labor-Framework zur Verwendung des CAVE. Entwicklung einer Projektaufgabenstellung und Durchführung des Projektes zu Themengebieten der fortgeschrittenen Virtuellen Realität. Praktikumsbegleitend finden kleine Vorlesungseinheiten zu bestimmten Themen, die für konkrete Projekte gebraucht werden, statt.

Lernergebnisse:

Durch eine in kleinen Gruppen entwickelte und durchgeführte Projektaufgabe erlangen Studierende Kenntnisse über Spezialalgorithmen der Virtuellen Realität, die für gewöhnlich in vorgefertigten Softwaretools verborgen sind (z. B. Kalibrierung, Verarbeitung von Videobildern oder 3D-Klang). Auf diese Weise kann ein tiefergehendes Verständnis über typische VR-Installationen erlangt werden.

Voraussetzungen und Empfehlungen:

Grundlagen der Mathematik, Voresung "Programmstrukturen 1", Vorlesung "Virtual and Augmented Reality"

Literatur:

- Doug A. Bowman, Ernst Kruijff, Joseph J. Laviola: 3D User Interfaces: Theory and Practice, Addison-Wesley Longman, 2004.
- Ralf Dörner, et al.: Virtual und Augmented Reality (VR/AR): Grundlagen und Methoden der Virtuellen und Augmentierten Realität, Springer Vieweg, 2013.

Studiengänge:

- Informatik Master of Science

◈ MM162 – Moderne Software-Architekturen

Verantwortliche:	Ulrich Hoffmann
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM039 – Moderne Software-Architekturen
Lernform:	Vorlesung
Prüfungsform:	Portfolio-Prüfung
Prüfungsdauer:	30 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	Sommersemester
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	30 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	24 Stunden
Aufwand während Semesterferien:	0 Stunden
Flexibel einteilbarer Aufwand:	32 Stunden
Gesamtaufwand:	*Gesamtaufwand 86 Stunden weicht signifikant von erwarteten 5.0 ECTS ab.*
Lehrende:	Ulrich Hoffmann

Studieninhalte:

- Virtualisierung
- Container
- Cloud Computing
- Serverless Computing
- grundledende verteilte Algorithmen
    - Konsensalgorithmen,
    - Synchronisationsalogrithmen
- Architekturmuster
    - Map/Reduce
    - Service Orientierte Architektur
    - Microservices
    - Pipelines
        - AI Pipelines
        - Deployment Pipelines
        - DevOps Pipelines
- Seminaristischer Unterricht
    - Kurzvortrag zu ausgewähltem Grundlagenthema
    - praktisches Projekt zu ausgewählter Aufgabenstellung
        - regelmäßige Projektbesprechungen
        - Wissenschaftlicher Vortrag zur Vorstellung der Projektergebnisse

Lernergebnisse:

Die Studierenden ...

- besitzen eingehende Kenntnisse über Struktur, Entwicklung, Bereitstellung und Betrieb großer Software-Systeme.
- haben die Fähigkeit umfangreiche Softwaresysteme bereitzustellen.
- haben die Fähigkeit eigene Softwaresysteme bereitzustellen.
- besitzen Kenntnisse und Fähigkeiten, große Softwaresystem automatisch überwachen zu lassen und Messungen an ihnen durchzuführen.
- Sie besitzen grundlegende Fähigkeiten aufgrund von Messergebnissen eine Optimierung und Skalierung von Softwaresystemen durchzuführen.

- haben die Fähigkeit, für ein Softwaresystem Anforderungen zu erheben, es geeignet zu strukturieren und auf angemessene Weise in Betrieb zu nehmen.
- besitzen die Fähigkeit ihr Vorgehen kritisch zu hinterfragen und nachvollziehbar zu kommunizieren.
- besitzen die Fähigkeit die von ihnen im Rahmen der Projektarbeit erarbeiteten Sachverhalte zu kondensieren und in angemessenen Vortragsstil und geeigneter Präsentationstechniken nachvollziehbar dazustellen. In freier Diskussion können sie sich über komplexe wissenschaftliche Sachverhalte der großen Softwaresysteme und ihres Betriebs auseinandersetzen.

Voraussetzungen und Empfehlungen:

- Erfahrung in der imperativen und objekt-orientierten Programmierung
- grundlegendes Verständnis der Sicherheitsanforderungen in der Informationstechnik
- Fähigkeit zur Implementierung von Sicherheitsmaßnahmen
- Kentnisse über die Funktionsweisen von Rechnetzen und des World Wide Webs / Internets.

Literatur:

- Bass, Weber, Zhu: DevOps: A Software Architect's Perspective, Addison Wesley 2015
- Bass, Kazman, Clements: Software Architecture in Practice, Addison Wesley, 2012
- Fowler: Patterns of Enterprise Application Architecture, Addison Wesley, 2002
- Hapke, Nelson: Building Machine Learning Pipelines, O'Reilly Media, 2020
- Humble, Farley: Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation, Addison-Wesley 2010
- Josuttis: SOA in Practice: The Art of Distributed System Design, O'Reilly Media 2007
- Kim, Willis, Debois, Humble: The DevOPS Handbook, IT Revolution Press, 2016
- Krafzig, Banke, Slama: Enterprise SOA: Service-Oriented Architecture Best Practices, Prentice Hall 2004
- Newman: Monolith to Microservices, O'Reilly Media, 2019
- Saito, Lee, Wu: DevOps with Kubernetes, Packt Publishing, 2019

Studiengänge:

- Informatik Master of Science
- IT-Sicherheit Master of Science
- Wirtschaftsinformatik / IT-Management Master of Science

◈ MM197 – Konzepte des Machine Learning

Verantwortliche:	Christian-Arved Bohn
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM121 – Konzepte des Machine Learning
Lernform:	Vorlesung mit integrierter Übung
Prüfungsform:	Abnahme
Prüfungsdauer:	20 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	Sommersemester
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	24 Termine
Zeit in Veranstaltungen:	60 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	24 Stunden
Aufwand während Semesterferien:	12 Stunden
Flexibel einteilbarer Aufwand:	54 Stunden
Gesamtaufwand:	150 Stunden
Lehrende:	Ulrich Hoffmann

Studieninhalte:

Diese Vorlesung bietet eine fundierte Einführung in die theoretischen Grundlagen und praktischen Ansätze des Maschinellen Lernens. Ziel ist es, den Studierenden nicht nur die Anwendung bestehender Algorithmen zu vermitteln, sondern ein tiefes Verständnis für die zugrunde liegenden mathematischen und logischen Prinzipien zu entwickeln. Es wird hierbei der gesamten Lebenszyklus eines ML-Modells – von der Datenrepräsentation bis zur Evaluierung betrachtet.

Lernergebnisse:

Nach erfolgreichem Abschluss des Moduls verfügen die Studierenden über ein tiefgreifendes Verständnis der theoretischen Grundlagen des Machine Learning und können zwischen überwachten, unüberwachten sowie bestärkenden Lernparadigmen differenzieren. Sie besitzen die methodische Kompetenz, für spezifische Problemstellungen eigenständig geeignete Algorithmen auszuwählen und diese effizient zu implementieren. Sie sind in der Lage, Modelle durch gezieltes Hyperparameter-Tuning zu optimieren und deren Vorhersagegüte objektiv zu validieren.

Voraussetzungen und Empfehlungen:

Basiswissen in der Programmierung, insbesondere mittels Python, Grundlagen der Informatik und Statistik.

Literatur:

- Ian Goodfellow, Yoshua Bengio & Aaron Courville: Deep Learning
- Christopher M. Bishop: Pattern Recognition and Machine Learning
- Kevin P. Murphy: Probabilistic Machine Learning

Studiengänge:

- Data Science & Artificial Intelligence Master of Science
- Informatik Master of Science
- IT-Sicherheit Master of Science

- Wirtschaftsinformatik / IT-Management Master of Science

◈ MM010 – Aktuelle Entwicklungen in der Informatik

Verantwortliche:	Ulrich Hoffmann
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM031 – Workshop Aktuelle Entwicklungen in der Informatik
Lernform:	Workshop
Prüfungsform:	Portfolio-Prüfung
Prüfungsdauer:	30 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	30 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	70 Stunden
Aufwand während Semesterferien:	0 Stunden
Flexibel einteilbarer Aufwand:	50 Stunden
Gesamtaufwand:	150 Stunden
Lehrende:	Ulrich Hoffmann

Studieninhalte:

themenabhängig

Lernergebnisse:

Studierende besitzen ...

- die Fähigkeit sich intensiv mit jeweils aktuellen Themen der theoretischen, praktischen, angewandten Informatik auseinanderzusetzen.
- Kenntnisse ausgewählter, fortgeschrittenen, modernen Informatik-Themen. Sie kennen im Detail die im jeweiligen Themengebiet relevanten Fragestellungen und können Lösungsansätze im Hinblick auf ihre Eignung bewerten und beurteilen.
- die Fähigkeiten einschlägige Softwaresysteme der jeweiligen Themenstellung zu bewerten und einzusetzen.

Voraussetzungen und Empfehlungen:

Grundlegende Fertigkeiten der Online-Recherche einschließlich des Beurteilens von relevanten Trends mit Abgrenzung zum Buzzword-Bingo ist hilfreich. Das Lesen und Verstehen von Fachliteratur in englischer Sprache ist notwendig.

Literatur:

themenabhängig

Studiengänge:

- Informatik Master of Science

◈ MM027 – Konzepte der Datenbanktechnologie

Verantwortliche:	Dennis Proppe
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM002 – Konzepte der Datenbanktechnologie
Lernform:	Vorlesung
Prüfungsform:	Klausur / Mündliche Prüfung
Prüfungsdauer:	60 Min.
ECTS:	3.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	2 Semesterwochenstunden
Termine im Semester:	*Häufigkeit nicht angegeben.*
Zeit in Veranstaltungen:	*Kontaktzeit kann nicht ermittelt werden (braucht SWS und Häufigkeit).*
Sonstiger Arbeitsaufwand während Vorlesungszeit:	*Nicht angegeben.*
Aufwand während Semesterferien:	*Nicht angegeben.*
Flexibel einteilbarer Aufwand:	*Nicht angegeben.*
Gesamtaufwand:	*Gesamtaufwand kann nicht ermittelt werden (braucht SWS und Häufigkeit, ggf. müssen die übrigen Zeiten explizit auf 0 gesetzt werden).*
Lehrende:	Dennis Proppe Ulrich Hoffmann

Teilleistung:	TM003 – Übg. Konzepte der Datenbanktechnologie
Lernform:	Übung
Prüfungsform:	Abnahme
Prüfungsdauer:	10 Min.
ECTS:	2.0
Benotung:	Bestanden/nicht Bestanden
Turnus:	jährlich
Dauer (pro Termin):	2 Semesterwochenstunden
Termine im Semester:	*Häufigkeit nicht angegeben.*
Zeit in Veranstaltungen:	*Kontaktzeit kann nicht ermittelt werden (braucht SWS und Häufigkeit).*
Sonstiger Arbeitsaufwand während Vorlesungszeit:	*Nicht angegeben.*
Aufwand während Semesterferien:	*Nicht angegeben.*
Flexibel einteilbarer Aufwand:	*Nicht angegeben.*
Gesamtaufwand:	*Gesamtaufwand kann nicht ermittelt werden (braucht SWS und Häufigkeit, ggf. müssen die übrigen Zeiten explizit auf 0 gesetzt werden).*
Lehrende:	Tim Wetzel

Studieninhalte:

- Grundlagen Datenbanksysteme
  - Persistenz
  - Transaktionen
  - 2PL
  - Datenschutz und Datensicherheit
- Objekt-relationales Mapping
  - Java Persistence API (JPA)
- NoSQL-Datenbanksysteme
  - Verteilte Wert/Schlüssel-Speicher
  - Dokumentendatenbanken
  - Graph-Datenbanken
- Verteilung von Daten

Vorlesungsbegleitende praktische Übungen zu Objektrelationalem Mapping und anderen alternativen Persistenzansätzen.

Erstellung einer NoSQL-Datenbank mit einem kompletten CRUD-Zyklus.

Lernergebnisse:

Studierende ...

- beherrschen die Fähigkeit Objektrelationales Mapping anzuwenden bzw. in Betrieb zu nehmen und es zur Lösung von Problemen einzusetzen.
- sind mit den praktisch auftretenden Schwierigkeiten vertraut und können sie systematisch überwinden.
- sind in der Lage eine NoSQL-Datenbank einzurichten, sie mit Daten zu füllen und anfragen an sie zu stellen

---

Die Studierenden erlangen die ...

- Kenntnis, der für die Implementierung von Datenbanksystemen wichtigen Architekturprinzipien, Datenstrukturen und Algorithmen und damit Kenntnis des Aufbaus und der internen Arbeit eines großen komplexen Softwaresystems.
- Fähigkeit, die Arbeitsweise von Datenbanksystemen zu optimieren bzw. selbst Architekturen für große komplexe Softwaresysteme zu entwerfen.
- Fähigkeiten eines Datenbankadministrators für Datenbanksysteme.
- Konzepte und Techniken des Datenschutzes, als auch der Datensicherheit

Voraussetzungen und Empfehlungen:

- Vertrautheit mit den grundlegenden Konzepten von Datenbanksystemen, einschließlich der Prinzipien der Persistenz und Transaktionen.
- Verständnis der relationalen Datenbankmodelle und Kenntnisse in SQL.
- Kenntnisse in der objekt-orientierten Programmierung

Literatur:

- siehe Vorlesung
- diverse Online-Quellen

- KEMPER, Alfons; EICKLER, Andre:
  Datenbanksysteme - Eine Einführung. Oldenbourg Verlag, 2004
- KEITH, Mike; SCHINCARIOL, Merrik:
  Pro JPA 2 - Mastering the Java Persistence API. APress, 2009
- BAUER, Christian; KING, Gavin:
  Java Persistence with Hibernate,
  Manning, Greenwich, 2007
- SQL- & NoSQL-Datenbanken – Andreas Meier, Michael Kaufmann; eXamen.press Springer Vieweg
- Sieben Wochen, sieben Datenbanken – Eric Redmond, Jim R. Wilson; O‘Reilly
- NoSQL for Dummies, Adam Fowler; For Dummies-Verlag
- div. Konferenzbeiträge und Forschungsarbeiten zu moderneren Entwicklungen der Datenbanktechnologie

Studiengänge:

- Data Science & Artificial Intelligence Master of Science
- Informatik Master of Science
- IT-Sicherheit Master of Science
- Wirtschaftsinformatik / IT-Management Master of Science

◈ MM029 – Berechenbarkeit und Verifikation

Verantwortliche:	Sebastian Iwanowski
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch/englisch

Bestandteile:

Teilleistung:	TM033 – Berechenbarkeit und Komplexität, Formale Spezifikation und Verifikation
Lernform:	Vorlesung
Prüfungsform:	Klausur / Mündliche Prüfung
Prüfungsdauer:	120 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	Wintersemester
Dauer (pro Termin):	6 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	45 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	45 Stunden
Aufwand während Semesterferien:	60 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	150 Stunden
Lehrende:	Ulrich Hoffmann Sebastian Iwanowski

Studieninhalte:

- Berechenbarkeit und Nichtberechenbarkeit
  - Präzisierung der Begriffe Problem und Algorithmen für die Theorie der Berechenbarkeit
  - Turingmaschinen im Detail
  - Komplexitätsklassen für Turingmaschinen
  - Beispiele für unentscheidbare Probleme
  - Beweise der Unentscheidbarkeit für ausgewählte Probleme
- NP-vollständige Probleme
  - Historie des P-NP-Problems
  - Beweis der NP-Vollständigkeit von SATISFIABILITY
  - Übersicht über NP-vollständige Probleme
  - Reduktionsmethode zum Beweis von NP-Vollständigkeit mit Beispielen
- Optimierungsaufgaben für NP-vollständige Probleme
  - Lösungstechniken für NP-vollständige Probleme
  - Übersicht über wichtige Anwendungen - Vergleich zu Verfahren der Künstlichen Intelligenz

---

- Mathematische und logische Grundlagen der Spezifikation und Verifikation; Mengen, Multimengen, Verbände, partielle und totale Funktionen, algebraische Strukturen, Aussagen- und Prädikatenlogik, Modallogik, temporale Logik
- Algebraische Spezifikation; Terme, Gleichungen; Fallbeispiel einer algebraischen Spezifikation; Datenstrukturen, Operationen, Nachweis von Eigenschaften; maschinenunterstütztes Beweisen von Eigenschaften
- Modellorientierte Spezifikation; Fallbeispiel einer modellorientierten Spezifikation
- Konstruktion korrekter Programme aus Spezifikationen
- Aktuelle Spezifikationssprachen im Überblick

Lernergebnisse:

Nach Abschluss der Veranstaltung besitzen die Studierenden folgende Kompetenzen:

- Fundierter theoretischer Überblick über die Möglichkeiten des Problemlösens.
- Theoretisch fundierte Kenntnis der Grenzen der Berechenbarkeit und der effizienten Lösbarkeit.
- Kenntnis der Alternativen für die Praxis bei theoretisch unbefriedigenden Resultaten.

Die Studierenden ...

- erlangen fundierte Kenntnisse der mathematischen Grundlagen der formalen Spezifikation und Verifikation.
- beherrschen verschiedene Spezifikationsstile.
- bekommen einen Einblick in verschiedene formale Spezifikationssprachen.
- erlangen die Fähigkeit, Spezifikationen systematisch zu konstruieren.
- können mathematische Beweise von Eigenschaften spezifizierte Software-Systeme führen.
- erlangen grundlegende Kenntnisse der Verifikation mit automatischen Beweissystemen.
- sind mit der Spezifikation von Sicherheitsbedingungen vertraut.
- können beurteilen, wie sich formale Methoden auf die IT-Sicherheit auswirken.

Voraussetzungen und Empfehlungen:

Diskrete Mathematik aus dem Bachelorstudium, im Besonderen die Mengenlehre und Beweise

Literatur:

- Garey, Michael R.; Johnson, David S. (1979), Computers and Intractability: A Guide to the Theory of NP-Completeness, W. H. Freeman, ISBN 0-7167-1045-5
- Hopcroft, John E.; Motwani, Rajeev; Ullman, Jeffrey D.:
  Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie.
  2. überarb. Aufl. München: Addison-Wesley Longman Verlag, 2002.
- Vossen,Gottfried; Witt, Kurt-Ulrich:
  Theoretische Informatik.
  Braunschweig: Verlag Vieweg & Teubner 2004 (3. Auflage), ISBN 978-3528231477
- Wagenknecht, C.:
  Algorithmen und Komplexität,
  Fachbuchverlag Leipzig 2003
- Winter, R.:
  Theoretische Informatik,
  Oldenbourg-Verlag München 2002

- BJØRNER, Dines:
  Software Engineering 1.
  Heidelberg: Springer Verlag, 2006
- DILLER, Antoni:
  Z An Introduction to Formal Methods.
  New York: Wiley & Sons, 1994
- EHRICH/GOGOLLA/LIPECK:
  Algebraische Spezifikation abstrakter Datentypen.
  Stuttgart: Teubner Verlag, 1989
- GOOS, Gerhard:
  Vorlesungen über Informatik Band 1 - Grundlagen und funktionales Programmieren.
  Heidelberg: Springer Verlag, 2005
- LAMPORT, Leslie:
  Specifying Systems.
  Amsterdam: Addison-Wesley, 2002
- SCHÖNING, Uwe:
  Logik für Informatiker.
  Heidelberg: Spektrum Akademischer Verlag, 2000
- WORDSWORTH, J., B.:
  Software Development with Z.
  New York: Addison-Wesley, 1992

Studiengänge:

- Informatik Master of Science
- IT-Sicherheit Master of Science

◈ MM035 – Distributed Systems

Verantwortliche:	Ulrich Hoffmann
Moduldauer:	6 Monate
Unterrichtssprache:	english

Bestandteile:

Teilleistung:	TM006 – Distributed Systems
Lernform:	Vorlesung
Prüfungsform:	Klausur / Mündliche Prüfung
Prüfungsdauer:	60 Min.
ECTS:	3.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	2 Semesterwochenstunden
Termine im Semester:	*Häufigkeit nicht angegeben.*
Zeit in Veranstaltungen:	*Kontaktzeit kann nicht ermittelt werden (braucht SWS und Häufigkeit).*
Sonstiger Arbeitsaufwand während Vorlesungszeit:	*Nicht angegeben.*
Aufwand während Semesterferien:	*Nicht angegeben.*
Flexibel einteilbarer Aufwand:	*Nicht angegeben.*
Gesamtaufwand:	*Gesamtaufwand kann nicht ermittelt werden (braucht SWS und Häufigkeit, ggf. müssen die übrigen Zeiten explizit auf 0 gesetzt werden).*
Lehrende:	Ulrich Hoffmann

Teilleistung:	TM007 – Tutorial: Distributed Systems
Lernform:	Übung
Prüfungsform:	Abnahme
Prüfungsdauer:	5 Min.
ECTS:	2.0
Benotung:	Bestanden/nicht Bestanden
Turnus:	jährlich
Dauer (pro Termin):	2 Semesterwochenstunden
Termine im Semester:	*Häufigkeit nicht angegeben.*
Zeit in Veranstaltungen:	*Kontaktzeit kann nicht ermittelt werden (braucht SWS und Häufigkeit).*
Sonstiger Arbeitsaufwand während Vorlesungszeit:	*Nicht angegeben.*
Aufwand während Semesterferien:	*Nicht angegeben.*
Flexibel einteilbarer Aufwand:	*Nicht angegeben.*
Gesamtaufwand:	*Gesamtaufwand kann nicht ermittelt werden (braucht SWS und Häufigkeit, ggf. müssen die übrigen Zeiten explizit auf 0 gesetzt werden).*
Lehrende:	Ulrich Hoffmann

Studieninhalte:

- Praktische Beispiele
- Allgemeine Anforderungen an verteilte Systeme
- Die Client-Server-Beziehung und daraus resultierende Fragen
- Kommunikation in verteilten Systemen
- Dienste benennen
- Techniken für Gleichzeitigkeit
- Ferngespräche
- Alternative Paradigmen (Akteurskonzept, ...)
- Synchronisierung von Daten und Prozessen
- Koordinationsmethoden
- Replikationstechniken
- WEB-Dienste mit SOAP und REST
- Fehlertoleranzkonzepte
- Sicherheit in verteilten Systemen
- Programmierung mit Threads
- Kommunikation über Sockets, Struktur von Clients und Servern
- Ferner Prozeduraufruf / entfernter Methodenaufruf
- Verwendung von Benennungsdiensten
- Programmierung von WEB-Diensten (SOAP, Server/Client, WSDL, Datenbindung)
- verteiltes Programmieren mit alternativen Konzepten
- Programmierung von Synchronisierungsalgorithmen
- Programmierung verteilter Wahlalgorithmen
- Programmierung von REST-basierten Dienstleistungen und Kunden
- Fehlertolerante Programmierung in verteilten Systemen

Vorlesung mit begleitenden praktischen Übungen zur Programmierung verteilter Systeme und ihrer Algorithmen in verschiedenen Programmierparadigmen.

Lernergebnisse:

Die Studierenden gewinnen ...

- gründliches Verständnis der Prinzipien verteilter Anwendungen.
- Kenntnisse in der Beherrschung von Basistechnologien und aktuellen Software-Werkzeugen für verteilte Systeme.
- Zustandskenntnis der sind in verschiedenen Anwendungsbereichen wie Dienstleistungsvermittlung und E-Commerce.
- Kenntnisse über IT-Sicherheitsfragen in verteilten Systemen sowie über verschlüsselter Kommunikation.
- Kenntnisse der grundlegenden Algorithmen in verteilten Systemen.
- genaue Kenntnis der aktuellen Web-Service-Architekturen.
- praktische Fähigkeiten zur Realisierung eines Projekts.
- verteilte Programmierkenntnisse in verschiedenen Paradigmen.

Die Studenten ...

- erlangen die Fähigkeit, typische Softwaresysteme (Middleware) im Bereich der verteilten Systeme zu bedienen und zur Problemlösung einzusetzen.
- sind an Probleme gewöhnt, die in der Realität auftreten, und in der Lage, diese zu überwinden.
- haben einige praktische Erfahrungen mit IT-Sicherheitsfragen.
- wissen, wie man Verschlüsselung in verteilten Umgebungen einsetzt.
- eignen sich durch praktische Erfahrung ein tiefes Wissen über die spezifischen Eigenschaften verteilter Systeme an. Sie können diese Eigenschaften kategorisieren und bewerten.

Voraussetzungen und Empfehlungen:

- Erfahrung in der imperativen und objekt-orientierten Programmierung
- grundlegendes Verständnis der Sicherheitsanforderungen in der Informationstechnik
- Fähigkeit zur Implementierung von Sicherheitsmaßnahmen
- Kentnisse über die Funktionsweisen von Rechnetzen und des World Wide Webs / Internets.

Literatur:

- siehe Vorlesung
- Zahlreiche Online-Ressourcen

- ARMSTRONG, Joe:
  Programming Erlang.
  Pragmatic Programmers, 2007
- ODERSKY, Martin; SPOON, Lex; VENNERS, Bill:
  Programming in Scala.
  Artima Press, Mountain View, 2008
- COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim:
  Distributed Systems, Concepts and Design.
  Addison-Wesley, 2011, ISBN 0-1321-4301-1
- TANENBAUM, Andrew; VAN STEEN, Marten:
  Distributed Systems, Priciples and Paradigms.
  Prentice Hall, 2006, ISBN 0-1323-9227-5

Studiengänge:

- Informatik Master of Science
- IT-Sicherheit Master of Science
- IT Engineering Master of Science

- Wirtschaftsinformatik / IT-Management Master of Science

◈ MM042 – Digitale Kommunikationssysteme und Reconfigurable Computing

Verantwortliche:	Sergei Sawitzki
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM034 – Digitale Kommunikationssysteme
Lernform:	Vorlesung
Prüfungsform:	Mündliche Prüfung
Prüfungsdauer:	30 Min.
ECTS:	1.0
Benotung:	Drittelnoten
Turnus:	jährlich
Dauer (pro Termin):	2 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	15 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	12 Stunden
Aufwand während Semesterferien:	1 Stunden
Flexibel einteilbarer Aufwand:	2 Stunden
Gesamtaufwand:	30 Stunden
Lehrende:	Sergei Sawitzki

Teilleistung:	TM035 – Prakt. Reconfigurable Computing, Reconfigurable Computing
Lernform:	Praktikum
Prüfungsform:	Mündliche Prüfung
Prüfungsdauer:	30 Min.
ECTS:	4.0
Benotung:	Drittelnoten
Turnus:	Wintersemester
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	4 Termine
Zeit in Veranstaltungen:	10 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	2 Stunden
Aufwand während Semesterferien:	5 Stunden
Flexibel einteilbarer Aufwand:	103 Stunden
Gesamtaufwand:	120 Stunden
Lehrende:	Sergei Sawitzki

Studieninhalte:

- Einführung und Begriffswelt
  - Rekonfigurierbares Rechnen, Rechnerparadigmen
  - ASIC, ASIPS, Mikroprozessoren, FPGAs und ihre funktionale Dichte
  - Einordnung und Klassifizierung der rekonfigurierbaren Systeme
- Schaltungstechnische Basis rekonfigurierbarer Systeme
  - PAL, PLA, PLD
  - CPLD und FPGA
  - hybride Systeme
- Entwurfsfluss und Besonderheiten
  - Hardwareentwurf, Entwurfsschritte
  - Retargierbares Übersetzen
  - Hardware / Software-Codesign
- Anwendungen und Anwendungsentwicklung
  - Klassifizierung
  - Umsetzung
  - Einbindung rekonfiguriebarer Hardware und Kommunikationskonzepte
  - Schnittstellen und Betriebssysteme
- Fortgeschrittene Techniken
  - Dynamische Rekonfiguration
  - Partielle Rekonfiguration
  - Selbstmodifizierende Architekturen
  - System-on-reconfigurable-chip
- Systembeispiele und Fallstudien
  - ISA-orientierte Architekturen
  - Lose gekoppelte Architekturen
  - Datenfluss-Architekturen
- Signale
  - Klassifikation und Analyse
  - Fourier-Transformation
  - Zeit, Frequenz und Bandbreite
- Modulation
  - Formatierung
  - Basisband-Modulation
  - Trägermodulation

    - Impulsformung
- Kanalkodierung
    - Block-Kodes
    - Faltungskodes
    - Iterative Kodierungsverfahren
    - Kodespreizung und -kaskadierung
- Frequenzspreizung und Multiplexverfahren
    - Grundlagen
    - Frequenzspreizung
    - Multiplexverfahren
    - Vielträgermodulation, OFDM-Systeme
- Systemstudien (z. B. wahlweise W-USB, WLAN, DOCSIS oder andere)
- Vorstellung der Aufgabenstellung
- Einarbeitung in die Entwurfswerkzeuge
- Umsetzung und Dokumentation der Aufgabe

Lernergebnisse:

Die Studierenden ...

- besitzen eine vertiefte Kenntnis moderner Übertragungssysteme, insbesondere des Aufbaus und der Funktionsweise von Basisband Transceivern
- kennen verschiedene Implementierungsaspekte von digitalen Kommunikationssystemen
- verstehen die Abhängigkeiten zwischen verschiedenen Systemparametern und der erreichbaren Übertragungsqualität
- verstehen Qualitätskriterien digitaler Kommunikationssysteme und Einflussfaktoren bei digitaler Datenübertragung
- sinnd befähigt digitale Übertragungsstandards zu interpretieren und auf der Ebene der Systemarchitektur (bis hin zur algorithmischen Ebene) zu spezifizieren und zu entwerfen
- kennen rekonfigurierbare Rechnersysteme als Entwurfsvariante des modernen Systementwurfs
- kennen die schaltungstechnische Basis und Hardware-Plattformen des Reconfigurable Computing
- sind befähigt, Vor- und Nachteile einer rekonfigurierbaren Implementierung eines Systems realistisch abschätzen zu können
- können eine Anwendung mit Hilfe rekonfigurierbarer Hardware implementieren

Voraussetzungen und Empfehlungen:

- grundlegende Kenntnisse analoger und digitaler Signalverarbeitung
- Kenntnisse der Rechnerarchitektur
- Programmierkenntnisse (vorzugsweise C)
- VHDL-Kenntnisse

Literatur:

- Lüders, Christian: Mobilfunksysteme, Vogel Verlag 2001
- Pehl, Erich: Digitale und analoge Nachrichtenübertragung, Hüthig Verlag 2001
- Werner, Martin: Nachrichtentechnik, Vieweg Verlag 2002
- Read, Richard: Nachrichten und Informationstechnik, Pearson Studium 2004
- Dankmeier, Wilfried: Grundkurs Codierung, Vieweg Verlag 2006
- Tietze, Ulrich; Schenk, Christoph: Halbleiterschaltungstechnik, 15. Auflage, Springer Verlag, 2016
- Sklar, Bernard: Digital Communications. Fundamentals and Applications, 2nd~ edition, Prentice Hall, 2001
- Bobda, Christophe: Introduction to Reconfigurable Computing: Architectures, algorithms and applications, Springer 2007

- Hauck, Scott; DeHon, Andre: Reconfigurable computing: the theory and practice of FPGA-based computation, Morgan Kaufmann Publishers 2008
- Hsiung, Pao-Ann; Santambrogio, Huang, Chun-Hsian: Reconfigurable System Design and Verification, CRC Press 2009

Aufgabenabhängig können weitere anwendungsspezifische Quellen herangezogen werden (z., B. Bildverarbeitung, Kryptographie, digitale Signalverarbeitung usw.)

- Bobda, Christophe: Introduction to Reconfigurable Computing: Architectures, algorithms and applications, Springer 2007
- Hauck, Scott; DeHon, Andre: Reconfigurable computing: the theory and practice of FPGA-based computation, Morgan Kaufmann Publishers 2008
- Hsiung, Pao-Ann; Santambrogio, Huang, Chun-Hsian: Reconfigurable System Design and Verification, CRC Press 2009

Studiengänge:

- Informatik Master of Science
- IT Engineering Master of Science

◈ MM044 – Fotorealismus und Simulation

Verantwortliche:	Christian-Arved Bohn
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM036 – Fotorealismus und Simulation, Visualisierung
Lernform:	Vorlesung mit integrierter Übung
Prüfungsform:	Mündliche Prüfung
Prüfungsdauer:	20 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	Wintersemester
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	30 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	24 Stunden
Aufwand während Semesterferien:	24 Stunden
Flexibel einteilbarer Aufwand:	72 Stunden
Gesamtaufwand:	150 Stunden
Lehrende:	Christian-Arved Bohn

Studieninhalte:

Über die Grundlagen der Computeranimation hinaus werden in der Veranstaltung fortgeschrittene Algorithmen der Animation und Visualisierung besprochen. Der erste Teil der Veranstaltung behandelt unterschiedliche Modellierungstypen wie Fraktale, Perlin Noise, Lindenmayersysteme, implizite Modellierung, prozedurale Modellierung und "Articulated Figures". Der zweite Teil erörtert Reaktions-Diffusions-Modelle zur Simulation von Flüssigkeiten und Gasen und zur Modellierung von Stoffen und Haaren - abgerundet durch eine Übersicht über klassische Methoden der Visualisierung von Strömungsvorgängen.

Studierende erhalten einen umfassenden Überblick über aktuelle Techniken der Simulation von Lichtausbreitung, um virtuelle Szenen realitätsnah darzustellen. Der erste Teil der Veranstaltung ist eine Einführung in die Radiometrie. Darauf basierend folgt der grundlegende Algorithmus des Monte Carlo Path Tracing bzw. des Stochastic Path Tracing mit diversen Optimierungsmethoden, wie z. B. Stochastic Ray Tracing versus Stochastic Light Tracing, verschiedene Sampling-Methoden und dem Algorithmus des Photon Mapping.

Mit diesem Wissen über das ideale Lichtmodell werden klassische Methoden aufbereitet (z. B. Radiosity) und deren physikalische Grundlage in der Radiometrie beleuchtet. Der Fokus des zweiten Teils der Veranstaltung ist die Kinetik, d.h. die Bewegung fester Körper unter Einwirkung von Kräften. Die Berechnung dieser bzw. die Rigid Body Simulation wird physikalisch und im Hinblick auf die Verwendung in Computerspielen betrachtet, bei der es darum geht, numerische Probleme der Simulation so zu lösen, dass die Echtzeitberechnung möglich ist. Die Rigid Body Simulation ist Basis für die realistische Bewegung von Körpern virtuellen Szenen.

Lernergebnisse:

Studierende erlangen das Verständnis

- über Algorithmen der Visualisierung und
- über fortgeschrittene Algorithmen der Computeranimation.

Studierende erhalten die Fähigkeit zur Einschätzung der Bedeutung der physikalischen Simulation. Verständnis der physikalischen Simulation in der Computergrafik, insbesondere die Simulation von

Lichttransfer und Kinetik.

Voraussetzungen und Empfehlungen:

Grundlagen der Mathematik, Vektorrechnung, Lineare Algebra, Grundlagen der Computergrafik

Literatur:

- Henrik W. Jensen: Realistic Image Synthesis Using Photon Mapping, Peters, Wellesley, 2001.
- Ian Millington: Game Physics Engine Development, Morgan Kaufmann, 2007.
- Kenny Erleben et al.: Physics-Based Animation, Course Technology, 2005.

- Kenny Erleben et al.: Physics-Based Animation, Course Technology, 2005.
- Alan Watt, Mark Watt: Advanced Animation and Rendering Techniques, Addison Wesley Longman Limited, 1998.
- G. M. Nielson, H. Hagen, H. Müller: Scientific Visualization, IEEE, 1997.
- Charles D. Hansen, Chris R. Johnson: The Visualization Handbook, Academic Press Inc, 2004.

Studiengänge:

- Informatik Master of Science

◈ MM048 – Projekt Informatik

Verantwortliche:	Ulrich Hoffmann
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM037 – Projekt Informatik
Lernform:	Projektarbeit
Prüfungsform:	Schriftl. Ausarbeitung (ggf. mit Präsentation)
Prüfungsdauer:	30 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	jedes Semester
Dauer (pro Termin):	0 Semesterwochenstunden
Termine im Semester:	*Häufigkeit nicht angegeben.*
Zeit in Veranstaltungen:	*Kontaktzeit kann nicht ermittelt werden (braucht SWS und Häufigkeit).*
Sonstiger Arbeitsaufwand während Vorlesungszeit:	*Nicht angegeben.*
Aufwand während Semesterferien:	*Nicht angegeben.*
Flexibel einteilbarer Aufwand:	*Nicht angegeben.*
Gesamtaufwand:	*Gesamtaufwand kann nicht ermittelt werden (braucht SWS und Häufigkeit, ggf. müssen die übrigen Zeiten explizit auf 0 gesetzt werden).*
Lehrende:	Ulrich Hoffmann

Studieninhalte:

Themenstellungen aus den Arbeitsgruppen und Laboren der Hochschule und aus Kooperationsprojekten mit externen Firmen.
Das Projekt wird in der Hochschule bearbeitet. Die Themenstellungen sollen dabei möglichst interdisziplinär sein, also sowohl Informatik- als auch anwendungsbereichsspezifische Aspekte enthalten.

Lernergebnisse:

Selbständiges und eigenverantwortliches Einarbeiten in eine aktuelle Themenstellung aus der Informatik unter zur Hilfenahme aktueller Quellen aus dem wissenschaftlichen Umfeld.
Erkennen der Bedeutung des methodischen und wissenschaftlichen Arbeitens für die Sicherung der Qualität einer Software-Lösung.
Interdisziplinäres Arbeiten und Kommunikation mit Fachleuten aus Informatik-fremden Bereichen.
Praktische Erfahrungen sammeln im Projekt-Management und den Bereichen Projektplanung, Koordination, Aufgabenaufteilung, Zeitmanagement, Delegation und Controlling.
Stärkung der sozialen Kompetenzen in den Bereichen Teamarbeit, Selbstständigkeit, Eigenverantwortung und Selbstorganisation.

Voraussetzungen und Empfehlungen:

- Vertrautheit mit den grundlegenden Konzepten und Phasen des Projektmanagements (Initiierung, Planung, Durchführung, Abschluss).
- Verständnis der verschiedenen Projektmanagementmethoden (z.B. Wasserfall, Agile).
- Fähigkeit zur Motivation und zum flexiblen Reagieren auf Änderungen.
- Grundlegende Kenntnisse in der Informatik und objektorientierten Programmiertechniken, insbesondere im Kontext von Software-Projekten.

Literatur:

Themenabhängig

Studiengänge:

- Informatik Master of Science

◈ MM198 – Symbolische Künstliche Intelligenz

Verantwortliche:	Gerd Beuster
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch/englisch

Bestandteile:

Teilleistung:	TM122 – Symbolische Künstliche Intelligenz
Lernform:	Vorlesung
Prüfungsform:	Klausur / Mündliche Prüfung
Prüfungsdauer:	90 Min.
ECTS:	5.0
Benotung:	Drittelnoten
Turnus:	Wintersemester
Dauer (pro Termin):	4 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	30 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	120 Stunden
Aufwand während Semesterferien:	0 Stunden
Flexibel einteilbarer Aufwand:	0 Stunden
Gesamtaufwand:	150 Stunden
Lehrende:	Gerd Beuster

Studieninhalte:

- Einführung in die Künstliche Intelligenz
- Intelligente Agenten
- Suchverfahren
- Aussagenlogik
- Logikbasierte autonome Agenten
- Prädikatenlogik
- Formale Logik und Sprachmodelle
- Grenzen der Prädikatenlogik
- Logikprogrammierung
- Knowledge Graphs

Lernergebnisse:

Die Studierenden sind in der Lage, Probleme der realen Welt in die Formalismen der klassischen Logiken (Aussagen- und Prädikatenlogik) umzusetzen. Sie kennen die Syntax und Semantiken der klassischen Logiken und die Grenzen der formallogischen Beweisbarkeit. Sie sind mit Methoden des automatischen Schließens vertraut. Die Studierenden können große Sprachmodelle (LLMs) in Kombination mit formallogischen Methoden nutzen.

Voraussetzungen und Empfehlungen:

Das Modul setzt voraus, dass die Studierenden die grundlegenden Algorithmen der Informatik und Grundlagen diskreter algebraischer Strukturen kennen.

Literatur:

- Harrison, John: Handbook of Practical Logic and Automated Reasoning, Cambridge: Cambridge University Press, 2009.
- Mackworth, Alan K.; Poole, David: Artificial Intelligence : Foundations of Computational Agents. 3. Auflage. Cambridge: Cambridge University Press, 2023.
- Norvig, Peter; Russell, Stuart: Artificial Intelligence : A Modern Approach. 4. Auflage. München: Pearson Deutschland GmbH, 2021.

- Schöning, Uwe: Logik für Informatiker, 5. Auflage. Heidelberg: Spektrum Akademischer Verlag, 2000.
- Lipovaca, Miran: Learn You a Haskell for Great Good! San Francisco (CA), USA: No Starch Press, 2012.
- Blackburn, Patrick; Bos, Johan; Striegnitz, Kristina: Learn Prolog Now!. London, UK: College Publications, 2006.

Studiengänge:

- Data Science & Artificial Intelligence Master of Science
- Informatik Master of Science

◈ MM050 – Master-Thesis

Verantwortliche:	Sergei Sawitzki
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	MTH – Master-Thesis
Lernform:	Thesis
Prüfungsform:	Abschlussarbeit
Prüfungsdauer:	60 Min.
ECTS:	27.0
Benotung:	Zehntelnoten
Turnus:	jedes Semester
Dauer (pro Termin):	0 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	0 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	400 Stunden
Aufwand während Semesterferien:	400 Stunden
Flexibel einteilbarer Aufwand:	40 Stunden
Gesamtaufwand:	840 Stunden
Lehrende:	Sergei Sawitzki

Studieninhalte:

themenabhängig

Lernergebnisse:

Die Studierenden

- können komplexe Aufgabenstellungen selbständig zu erarbeiten
- können Problemstellungen im größeren Kontext zu verorten
- sind in der Lage wissenschaftliche Methoden für die Problemlösung einzusetzen
- können Ergebnisse überzeugend unter besonderer Berücksichtigung der Aspekte wissenschaftlichen Arbeitens darzustellen

Voraussetzungen und Empfehlungen:

Fachliche und persönliche Kompetenzen der zurückliegenden Semester, insbesondere themenabhängig fachverwandte Module

Literatur:

themenabhängig

Studiengänge:

- Betriebswirtschaftslehre Master of Science
- Data Science & Artificial Intelligence Master of Science
- E-Commerce Master of Science
- Informatik Master of Science
- IT-Sicherheit Master of Science
- IT Engineering Master of Science
- Sustainable & Digital Business Management Master of Science
- Wirtschaftsinformatik / IT-Management Master of Science

- Wirtschaftsingenieurwesen Master of Science

◈ MM058 – Master-Kolloquium

Verantwortliche:	Sergei Sawitzki
Moduldauer:	6 Monate
Unterrichtssprache:	deutsch

Bestandteile:

Teilleistung:	TM010 – Master-Kolloquium
Lernform:	Kolloquium
Prüfungsform:	Kolloquium
Prüfungsdauer:	60 Min.
ECTS:	3.0
Benotung:	Drittelnoten
Turnus:	jedes Semester
Dauer (pro Termin):	0 Semesterwochenstunden
Termine im Semester:	12 Termine
Zeit in Veranstaltungen:	0 Stunden
Sonstiger Arbeitsaufwand während Vorlesungszeit:	10 Stunden
Aufwand während Semesterferien:	10 Stunden
Flexibel einteilbarer Aufwand:	40 Stunden
Gesamtaufwand:	*Gesamtaufwand 60 Stunden weicht signifikant von erwarteten 3.0 ECTS ab.*
Lehrende:	Sergei Sawitzki

Studieninhalte:

- nach Thema der Master-Arbeit unterschiedlich
- Fachvortrag über Thema der Master-Thesis sowie über die gewählte Vorgehensweise und die Ergebnisse
- Diskussion der Qualität der gewählten Lösung
- Fragen und Diskussion zum Thema der Master-Arbeit und verwandten Gebieten

Lernergebnisse:

Die Studierenden ...

- besitzen die Fähigkeit der konzentrierten Darstellung eines intensiv bearbeiteten Fachthemas unter besonderer Berücksichtigung der Aspekte wissenschaftlichen Arbeitens
- verfestigen die Kompetenz, eine fachliche Diskussion über eine Problemlösung und deren Qualität zu führen
- verfügen über ausgeprägte Kommunikations- und Präsentationsfähigkeiten

Voraussetzungen und Empfehlungen:

Fachliche und persönliche Kompetenzen der zurückliegenden Semester, insbesondere themenabhängig fachverwandte Module und Master-Thesis

Literatur:

themenabhängig

Studiengänge:

- Betriebswirtschaftslehre Master of Science
- Data Science & Artificial Intelligence Master of Science
- E-Commerce Master of Science
- Informatik Master of Science

- IT-Sicherheit Master of Science
- IT Engineering Master of Science
- Sustainable & Digital Business Management Master of Science
- Wirtschaftsinformatik / IT-Management Master of Science
- Wirtschaftsingenieurwesen Master of Science