Simulationssystem zur Interaktion mit realen Fischen unter Verwendung von Analyse-durch-Synthese-Verfahren

DISSERTATION

zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

vorgelegt von

Dipl.-Inform. Klaus Müller

eingereicht bei der Naturwissenschaftlich-Technischen Fakultät

der Universität Siegen

Siegen 2019

Betreuer und erster Gutachter
Prof. Dr.-Ing. Klaus-Dieter Kuhnert
Universität Siegen

Zweite Gutachterin
Prof. Dr. Klaudia Witte
Universität Siegen

Tag der mündlichen Prüfung
06. Mai 2020

Diese Dissertation wurde im Rahmen des interdisziplinären Forschungsprojekts *Analyse durch Synthese mit virtuellen Fischen als neue Versuchsmethode in Untersuchungen zur Partnerwahl*, als Kooperation zwischen dem Institut für Biologie und dem Institut für Echtzeit Lernsysteme der Universität Siegen, durchgeführt. Das Forschungsprojekt wurde durch die Deutsche Forschungsgemeinschaft (DFG; KU 689/11-1 und WI 1531/12-1) gefördert.

Zusammenfassung

Der Begriff der virtuellen Realität (VR) ist in aller Munde und zu einem Inbegriff des Fortschritts geworden. Die meisten VR-Systeme sind für den menschlichen Betrachter konzipiert. In der vorliegenden Arbeit wird hingegen ein VR-System für Fische entwickelt, welches in der Verhaltensforschung Anwendung findet. Durch die Interaktion eines virtuellen mit einem realen Fischs werden zur Erforschung des Paarungsverhaltens neue Möglichkeiten geschaffen und bestehende Arbeitsabläufe vereinfacht.

Ziel der Arbeit ist die Entwicklung eines Versuchsstandes zur Durchführung von visuellen, interaktiven Verhaltensforschungsexperimenten, bei denen ein virtueller 3D-Fisch als Stimulus während der Partnerwahl eingesetzt wird und mit dem realen Testfisch interagiert.

Als Basis des Systems wurde im Rahmen dieser Arbeit eine System-Architektur entworfen, die aus der Robotik abgeleitet und auf den speziellen Einsatz für Versuchsstände zur visuellen Fisch-Computer-Interaktion angepasst wurde.

Das entwickelte Fisch-Animationssystem ermöglicht eine einfache Gestaltung und benutzerfreundliche Animation virtueller Fischstimuli. Um darüber hinaus eine Interaktion zu realisieren, wird die Position des realen Fischs verfolgt und die Fischbewegung exakt rekonstruiert. Das optische Positionsverfolgungssystem trackt die 3D-Position der Fische in Echtzeit und erzeugt eine einfache geometrische Rekonstruktion der Versuchstiere. Die entwickelte Methode zur Kompensation der am Aquarium auftretenden Lichtbrechung erhöht dabei die Positionsgenauigkeit. Zusätzlich werden Methoden zur automatischen Initialisierung, zur einfachen Kamerakalibrierung und zur Erhöhung der Systemzuverlässigkeit bei welliger Wasseroberfläche vorgestellt.

Zur Rekonstruktion der Fischbewegung kommen Analyse-durch-Synthese-Verfahren zum Einsatz, mit denen die geometrischen Eigenschaften des realen Fisches in Modellparameter des virtuellen Modells übertragen werden. Auf diese Weise können aufgenommene Bewegungssequenzen zum direkten Einsatz im Animationssystem aufbereitet werden. Zur Erhöhung der Präzision wurde ebenfalls eine Methode zur Kompensation der Lichtbrechung entwickelt.

Der Versuchsstand wurde erfolgreich in mehreren Experimenten mit Breitflossenkärpflingen an der Universität Siegen getestet und ist in Teilen der Forschungsgemeinschaft als quelloffene Software zur Verfügung gestellt worden.

Abstract

The concept of virtual reality (VR) has become very popular and the epitome of progress. Most VR systems are designed for humans. In the present work, on the other hand, a VR system for fish is developed, which is used in behavioral research. Through the interaction of a virtual with a real fish, new possibilities are created and existing workflows are simplified.

The aim of this work is the development of a test stand for visual, interactive behavioral research experiments. During this a virtual 3D fish is used as a stimulus during mate choice experiments and is able to interact with the real test fish.

As a basis, a system architecture was designed in the context of this work, which was derived from robotics and adapted to the special use of test rigs for visual fish-computer interaction. The developed fish animation system enables a simple way of design and an user-friendly animation of virtual fish stimuli. In order to realize an interaction, the position of the real fish is tracked and the fish movement is exactly reconstructed. The optical tracking system tracks the 3D position of the fish in real time and generates a simple geometric reconstruction of the experimental animals. The developed method for compensation of the light refraction occurring at the aquarium increases the position accuracy. In addition, methods for automatic initialization, simple camera calibration and for increasing system reliability with wavy water surfaces are presented.

For the reconstruction of the fish movement, analysis-by-synthesis methods are used, with which the geometric properties of the real fish are transferred into model parameters of the virtual model. In this way, recorded motion sequences can be processed for direct use in the animation system. In order to increase the precision, a method for the compensation of light refraction has also been developed for the reconstruction part.

The complete system has been successfully tested in several experiments with Sailfin mollies at the University of Siegen and has been partially made available as open source software to the research community.

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

Bereits im Jahre 1950 entwarf einer der Pioniere der Informatik, Alan Turing, einen Test, dessen Ziel es war, festzustellen, ob eine Maschine ein gleichwertiges Denkvermögen wie ein Mensch besitzt (künstliche Intelligenz). Der Maschine wurde dies bescheinigt, sobald ein Mensch in einer Chat-Kommunikation mit der Maschine auf der einen und einem weiteren Menschen auf der anderen Seite nicht unterscheiden konnte, bei welchem seiner beiden Kommunikationspartner es sich um die Maschine handelt. In der heutigen Zeit, in der die virtuelle und reale Welt mehr und mehr ineinandergreifen, würden viele, im Hintergrund agierende Maschinen, den Turing-Test bestehen: Telefon- und Chatbots beantworten unsere Fragen, Algorithmen stellen personalisierte und vermeintlich objektive Nachrichten zusammen oder schreiben ganze Zeitungsartikel eigenständig. Die Grenze zwischen realer und virtueller Welt sind fließender denn je.

Auch zur Erforschung des Tierreichs im Allgemeinen und zur Analyse von Tierverhaltensmustern im Speziellen eröffnet der Einsatz von intelligenten, virtuellen Systemen eine Vielzahl von neuen Möglichkeiten.

Ein Problem in der klassischen Verhaltensforschung ist die Reproduzierbarkeit von Verhaltensexperimenten, welche besonders für eine spätere statistische Auswertung essentiell ist. Dies stellt insbesondere dann eine Herausforderung dar, wenn das Experiment eine Interaktion mit einem weiteren Tier umfasst. Auch bei der Erforschung des Paarungsverhaltens von Fischen werden diese Probleme offensichtlich. Zu dessen Erforschung werden oft lebende Stimulus-

Fische (→ Stimulus (Verhaltensforschung)[1]) eingesetzt, die durch ihr äußeres Erscheinungsbild, ihr Verhalten oder die Interaktion mit Dritten, das Interesse des potenziellen Paarungspartners auf sich ziehen. Welche Faktoren letztendlich zur Partnerwahl beitragen, ist nach wie vor Gegenstand der Forschung.
Da es in Partnerwahlversuchen mit lebenden Stimulusfischen schwierig ist, Faktoren wie Erscheinungsbild, Verhalten und Interaktion zu beeinflussen, werden virtuelle, animierte 3D-Fische eingesetzt. Diese können sowohl in ihrer Erscheinung als auch in ihrem Verhalten beliebig modelliert werden. Eine neue Methode in diesem Feld ist der Einsatz von interaktiven 3D-Fischen, die in → Echtzeit auf die Bewegung des Testfisches reagieren und mit diesem interagieren können. Ein solches System umfasst mehrere Komponenten die aufeinander aufbauen: ein System zur Detektion, Verfolgung und Bewegungsrekonstruktion des lebenden Testfisches im dreidimensionalen Raum, ein Animationssystem zur Generierung der 3D-Videoanimation und ein Steuerungssystem, welches den virtuellen Fisch in Echtzeit auf die Bewegungen des Lebenden reagieren lässt. Nicht nur aus Sicht der Verhaltensforschung sondern auch aus Sicht der Informatik beinhaltet das Gesamtsystem viele spannende Forschungsfragen, die über den hier behandelten Kontext hinausreichen und sich auf viele weitere Anwendungsbereiche ausdehnen lassen.
Die im Jahre 1950 von Turing über die menschliche Chat-Kommunikation definierte künstliche Intelligenz, kann auf Basis des in dieser Arbeit entwickelten Systems auf die visuelle Kommunikation der Fische übertragen werden.

1.1 Ziel der Arbeit

Ziel dieser Arbeit ist die Erstellung einer Verfahrenskette zur Durchführung von interaktiven, animationsbasierten Experimenten zur Partnerwahl bei Fischen. Die Verfahrenskette umfasst Komponenten zur Erstellung der 3D-Stimulus-Modelle, zur Animation (manuell, halbautomatisch, automatisch und interaktiv), zur echtzeitfähigen 3D-Positionsverfolgung und zur Modell-basierten

[1] Durch die hier vorgestellte interdisziplinäre Forschung werden sowohl Termini aus der Informatik als auch aus der Biologie verwendet. Um die Lesbarkeit der Arbeit zu erhöhen werden diese nicht im Fließtext, sondern im Glossar am Ende der Arbeit vereinfacht erläutert. Im Text markiert ein vorangestellter Pfeil (→) den im Glossar erklärten Fachbegriff.

Bewegungsrekonstruktion mittels des Analyse-durch-Synthese Verfahrens. Alle genannten Komponenten sollen dabei auch für unerfahrene Benutzer einfach zu bedienen sein und einen hohen Grad an Automation mit sich bringen.
Zur Erstellung von 3D-Modellen ist Erfahrung im Umgang mit 3D-Modellierungssoftware vonnöten. Um auch fachfremden Benutzern die Erstellung und Anpassung von individuellen Fisch-Stimuli zu ermöglichen, muss die Software Methoden umfassen, die den Benutzer unterstützen und Verfahren zur einfachen Erstellung und Konfiguration bereitstellen.
Auch der Prozess der 3D-Animation erfordert viel Erfahrung, um authentische Bewegungssequenzen zu erstellen. Darum muss abweichend von den sonst üblichen Animationsverfahren eine auf den Anwendungsfall der Fischanimation spezialisierte Methode entwickelt werden, die einfach und intuitiv eine naturgetreue Animation für unerfahrene Benutzer ermöglicht. Dabei sind einige Besonderheiten zu beachten: eine erstellte Bewegungsabfolge muss stimulusunabhängig wiederzugeben sein. Dies hilft bei der Standardisierung der Versuche, da so dieselben Bewegungspfade mit unterschiedlichen Stimuli präsentiert werden können und das Augenmerk auf die eigentliche Forschungsfrage gelegt werden kann. Des Weiteren ist eine Methode zu erarbeiten, die den Experimentator während des Versuches unterstützt und die Animationen zur richtigen Zeit auf dem richten Anzeigengerät präsentiert.
Im Versuchsstand für die Partnerwahl bei Fischen ist nicht nur die manuelle, in einem Vorverarbeitungsschritt erstellte, und wiedergegebene Animation gefragt, sondern es muss zudem auch eine Möglichkeit zur automatischen Animation entwickelt werden, die auf äußere Signale reagieren kann.
Ein weiteres wichtiges Ziel der Arbeit ist die automatische und präzise 3D Positions- und Bewegungsschätzung der lebenden Fische im Aquarium. Die aus diesen Prozessen gewonnenen Daten werden entlang der kompletten Verfahrenskette benötigt: Zur Steuerung der interaktiven Fischanimation muss die 3D-Position des lebenden Fisches in Echtzeit bereitgestellt werden, um den virtuellen Fisch auf dessen Bewegung reagieren lassen zu können. Neben der Echtzeitfähigkeit muss zudem eine hohe Positionsgenauigkeit gewährleistet werden, da bei der interaktiven Animation der virtuelle Fisch dem echten Fisch zentimetergenau folgt; ein Positionsversatz würde den Effekt mindern. Zur Auswertung einer Versuchsreihe dienen die Tracking-Daten als Grundlage.

Sie geben Auskunft über Position, Geschwindigkeit und Bewegung der Fische während des Versuches.
Ein weiteres Ziel dieser Arbeit ist die automatische Überführung von Bewegungsdaten aus Videosequenzen auf 3D-Animationen. Ein solches Verfahren ermöglicht auf einfache Weise komplexe Bewegungen abzuleiten und diese während eines Versuches mittels 3D-Animation zu präsentieren.
Schlussendlich ist ein Gesamtsystem zu schaffen, in welches alle genannten Komponenten integriert werden können.

1.2 Wissenschaftlicher Beitrag

Die Arbeit ist in vier thematisch getrennte Bereiche unterteilt, deren wissenschaftlichen Beitrag im Folgenden erläutert wird.

1.2.1 System-Architektur für Versuchsaufbauten mit Fisch-Computer-Interaktion

Zwischen dem hier vorgestellten System zur Fisch-Computer-Interaktion und einem Robotik-System können viele Parallelen gezogen werden. Bezüglich der bisher entwickelten Software unterscheiden sich die Forschungsfelder jedoch eklatant: In der Robotik gibt es eine große, internationale Community, die kontinuierlich Software entwickelt und diese im Rahmen des OpenSource-Modells anderen Forschern zur Verfügung stellt. Im Bereich der Fisch-Verhaltensforschung gibt es im Vergleich nur sehr wenig Software-Entwicklungen, die zudem meistens nicht veröffentlicht werden.
In der hier vorgestellten Arbeit werden die beiden Welten vereint und Software aus der Robotik zur Erstellung eines Versuchsaufbaus der Fisch-Verhaltensforschung genutzt. Im Speziellen handelt es sich dabei um das Robotik-Framework *ROS (Robot Operating System)*, welches sich besonders durch Merkmale wie Skalierbarkeit, Erweiterbarkeit und Modularität für Software-Projekte in der Fisch-Verhaltensforschung auszeichnet. Des Weiteren bietet es viele verschiedene Werkzeuge, die auch für die Verhaltensforschung einen großen Mehrwert besitzen. Soweit dem Autor bekannt, wurde in diesem Projekt erstmals ein Robotik-Framework für die Fisch-Verhaltensforschung mit virtuellen Stimuli

genutzt und anhand des hier vorgestellten Systems die Integration beschrieben.
Zum Einsatz von Robotik-Frameworks in der Fisch-Verhaltensforschung wurde keine wissenschaftliche Publikation veröffentlicht. Allerdings bildete dieses die Grundlage für viele der im Rahmen dieses Projektes veröffentlichten Publikationen. Besonders [87] und [88] bauen im hohen Maße auf dem Robotik-Framework auf und nutzen dessen Werkzeuge und Infrastruktur.

1.2.2 Fisch-Tracking

Das Tracking von Fischposition und -bewegung wurde in zwei internationalen Publikationen veröffentlicht. In der ersten Veröffentlichung ([87]) wurde ein echtzeitfähiges 3D-Tracking-System vorgestellt, welches sich besonders durch drei Punkte von den bisherigen Systemen abgrenzt:

1. Die Lichtbrechung, die durch das Aquarium entsteht, wird durch Modellierung des Strahlengangs kompensiert, so dass eine hohe Präzision erreicht wird.

2. Das System ist in der Lage in Echtzeit die Konturen der Fische zu tracken und eine einfach 3D-Rekonstruktion zu erzeugen (Kastenfisch).

3. Das Verfahren ist zu einem hohen Grad automatisiert.

Der hohe Automatisierungsgrad wird sowohl während der Kamera-Kalibrierung als auch während des Tracking-Prozesses eingehalten. Die Kalibrierung wird automatisch anhand von farbigen Markierungen am Aquarium vor jedem Versuch durchgeführt. Für das Tracking-Verfahren wurde eine neue Methode entwickelt, die das Tracking-System in einem zweistufigen Prozess automatisch initialisieren kann.
Im Gegensatz zu bisherigen Kalibrierungsmethoden für Kameraaufbauten mit Aquarien, werden in der hier präsentierten Methode nicht nur die Kameraparameter, sondern auch die Lichtbrechungsebenen geschätzt. Dadurch ist eine hochgenaue Rekonstruktion der Lichtbrechung mittels Strahlverfolgungstechnik (→ Raytracing (Computergrafik)) möglich. Die hohe Präzision des Systems hilft zudem bei weiteren Problemlösungen wie Detektion von Spiegelungen und der 3D-Rekonstruktion von Objekten.

Das in der zweiten Veröffentlichung ([85]) präsentierte Verfahren basiert auf den zuvor veröffentlichten Methoden [87] und beschäftigt sich insbesondere mit einem Lösungsansatz zur ungeordneten Lichtbrechung auf Wasseroberflächen während des Trackings. Das Verfahren ist in der Lage die 3D-Position des Fisches auch bei einer welligen Wasseroberfläche in Echtzeit zu bestimmen.

Publikationen

[87] Müller, K. ; Schlemper, J. ; Kuhnert, L. ; Kuhnert, K.-D. : Calibration and 3D ground truth data generation with orthogonal camera-setup and refraction compensation for aquaria in real-time. In: International Conference on Computer Vision Theory and Applications (VISAPP) Bd. 3 IEEE, 2014, S. 626–634

[85] Müller, K. ; Gierszewski, S. ; Witte, K. ; Kuhnert, K.-D. : Where is my mate? Real-time 3-D fish tracking for interactive mate-choice experiments. In: 23rd International Conference on Pattern Recognition; VAIB 2016 - Visual observation and analysis of Vertebrate and Insect Behavior Workshop Proceedings, 2016

1.2.3 Virtuelle, interaktive Fisch-Stimuli zur Verhaltensforschung

Im Rahmen dieser Arbeit wurden mehrere innovative Methoden entlang der kompletten Verfahrenskette für die Verhaltensforschung mit virtuellen Fisch-Stimuli entwickelt, die im Folgenden genannt werden:

- erstes, dem Autor bekanntes, System zur interaktiven Stimulus-Animation im dreidimensionalen Raum
- Methode zur einfachen Generierung und Konfiguration von Fisch-Stimuli mit Hilfe einer intuitiven, grafischen Oberfläche [88]
- Verfahren zur halbautomatischen Stimulus-Animation mittels Gamecontroller [88, 126]
- System zur beliebigen Kombination von aufgenommenen Schwimmpfaden und Stimulus-Modellen [88]

- Verfahren zur automatischen Ablaufsteuerung eines Versuches [88]

Ein Teil des Systems wurde in [49] validiert und fand in [48] weitere Anwendung.

Publikationen

[88] Müller, K. ; Smielik, I. ; Hütwohl, J.-M. ; Gierszewski, S. ; Witte, K. ; Kuhnert, K.-D. : The virtual lover: variable and easily guided 3D fish animations as an innovative tool in mate-choice experiments with sailfin mollies-I. Design and implementation. In: Current Zoology 63 (2016), Nr. 1, S. 55–64. http://dx.doi.org/10.1093/cz/zow106. – DOI 10.1093/cz/zow106

[49] Gierszewski, S. ; Müller, K. ; Smielik, I. ; Hütwohl, J.-M. ; Kuhnert, K.-D. ; Witte, K. : The virtual lover: variable and easily guided 3D fish animations as an innovative tool in mate-choice experiments with sailfin mollies-II. Validation. In: Current Zoology 63 (2017), Nr. 1, S. 65–74. http://dx.doi.org/10.1093/cz/zow108. – DOI 10.1093/cz/zow108

[126] Smielik, I. ; Müller, K. ; Kuhnert, K.-D. : Fish motion simulation. In: European Simulation and Modelling Conference (ESM), 2015, S. 392–396

[48] Gierszewski, S. ; Baker, D. ; Müller, K. ; Hütwohl, J.-M. ; Kuhnert, K.-D. ; Witte, K. : Using the FishSim Animation Toolchain to Investigate Fish Behavior: A Case Study on Mate-Choice Copying In Sailfin Mollies. In: Journal of Visualized Experiments: JoVE (2018), Nr. 141. http://dx.doi.org/10.3791/58435. – DOI 10.3791/58435

1.2.4 Analyse-durch-Synthese zur Bewegungsrekonstruktion

Das System zur Posen- und Bewegungsrekonstruktion mittels Analyse-durch-Synthese leistet in mehrfacher Hinsicht einen Beitrag zur wissenschaftlichen Weiterentwicklung. Zur schnellen Initialisierung des Verfahrens wurde ein Algorithmus zur Auswahl von Merkmalsteilmengen entwickelt [89]. Bei diesem Verfahren werden anhand verschiedener Merkmalseigenschaften mehrere über

den Posenraum verteilte Merkmalsteilmengen bestimmt. Mit Hilfe des Verfahrens kann sowohl bei der Merkmalsextraktion als auch beim Abgleich Rechenzeit eingespart werden. Eine genaue Erklärung ist in Kapitel 5.4 zu finden.
Das zur Posen- und Bewegungsrekonstruktion entwickelte Verfahren nähert, auf Basis von Videosequenzen, Modellparameter eines oder mehrerer 3D-Fischmodelle an. Zur Erhöhung der Präzision wurde dem Synthese-System zusätzlich eine Methode zur Lichtbrechungs-Synthese hinzugefügt. Dies ist, soweit dem Autor bekannt, bisher einmalig in diesem Bereich. Das Gesamtsystem zeichnet sich zudem durch eine hohe Flexibilität aus: es können Videosequenzen einzelner oder mehrerer Kameras zur Analyse genutzt und einzelne oder mehrere Fische gleichzeitig analysiert werden. Das System wird in Kapitel 5 detailliert beschrieben.

Publikationen

[89] Müller, K. ; Smielik, I. ; Kuhnert, K.-D. : Optimal Feature-set Selection Controlled by Pose-space Location. In: International Conference on Computer Vision Theory and Applications (VISAPP) IEEE, 2016, S. 200–207

[86] Müller, K. ; Hütwohl, J.-M. ; Gierszewski, S. ; Witte, K. ; Kuhnert, K.-D. : Fish Motion Capture with Refraction Synthesis. In: Proceedings of 26. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision WSCG Bd. 26, 2018, S. 125–134

1.3 Aufbau der Arbeit

Die Arbeit ist in sechs Kapitel unterteilt. Da die Themenfelder der einzelnen Kapitel zwar thematisch zusammenhängen, jedoch jedes für sich einem anderen Forschungsbereich zugeordnet werden kann, enthält jedes Kapitel dieselbe Grundstruktur bestehend aus Einleitung, Stand der Technik und Fazit.
Kapitel 1 führt in die hier behandelte Thematik ein und gibt einen allgemeinen Überblick über die Arbeit. In Kapitel 2 wird die Architektur der gesamten Verfahrenskette erläutert. Dies ist besonders vor dem Hintergrund bedeutend, dass ein aus der Robotik stammendes →Framework für die hier entwickelte Verfahrenskette genutzt wird. Neben der Beschreibung des Frameworks ist

die Integration des Forschungsaufbaus in Kapitel 2 beschrieben. Des Weiteren werden Auswahlkriterien für die Hardwarekomponenten des Gesamtsystems definiert.
Das dritte und größte Kapitel dieser Arbeit beschäftigt sich mit dem Themenfeld des Fisch-Trackings. Zum besseren Verständnis umfasst dieses ein Unterkapitel, in dem die Grundlagen der späteren Verfahren erklärt werden. Der zweite Teil des Kapitels umfasst neben den eigentlichen Methoden zum Tracking auch Methoden zur Kalibrierung und Detektion, die als vorgelagerte Schritte des Trackings erforderlich sind.
Im vierten Kapitel wird der Themenbereich der virtuellen Fische behandelt. Neben der Erstellung der Stimulus-Fische werden die Methoden zur halbautomatischen, automatischen und interaktiven Animation sowie zur Präsentation der Stimuli beschrieben.
Das fünfte Kapitel beschreibt das Verfahren zur Bewegungsrekonstruktion auf Basis der Analyse-durch-Synthese Methode. Bei diesem werden die in Kapitel 4 entwickelten Modelle zur Rekonstruktion der Fischposen aus Kamerabildern genutzt. Die Posen werden mit Hilfe eines Optimierungsprozesses angenähert. Alle dazu erforderlichen Verfahrensschritte werden näher erläutert. Dazu zählt zu Beginn des Prozesses die Initialisierung der Modellparameter, um dem Optimierungsverfahren bestmögliche Startbedingungen zu gewährleisten. Nach der Initialisierung ist eine Metrik zum Vergleich des synthetischen und des realen Bildes nötig. Des Weiteren werden Strategien zum Umgang mit Verdeckung und Lichtbrechung eingeführt.
Im abschließenden Kapitel werden die zentralen Verfahren dieser Arbeit reflektiert und ein Ausblick auf mögliche anschließende Forschungsaktivitäten gegeben.

Kapitel 2

Hardwareauswahl und Systemarchitektur für Versuchsaufbauten mit Fisch-Computer-Interaktion

Ein Versuchsstand zur Interaktion zwischen virtuellen-computergesteuerten und realen Fischen umfasst mehrere Komponenten, die in Echtzeit miteinander kommunizieren müssen. Dazu zählt ein Fisch-Animationssystem, ein Sensorsystem zur Überwachung der realen Fische, eine Steuerung, welche die Bewegungsabläufe und somit die Interaktion zwischen den realen und virtuellen Fischen regelt und eine Bedienerschnittstelle zur Überwachung und Konfiguration des Systems. Ein ähnliches Anforderungsprofil findet man bei Robotersystemen. Auch diese benötigen Sensorsysteme, Steuerungen und Bedienerschnittstellen, die in Echtzeit miteinander kommunizieren müssen. Im Gegensatz zur Forschung und Entwicklung interaktiver, virtueller Tier-Versuchsstände gibt es im Bereich der Robotik eine aktive und breite Forschungsgemeinschaft, in der Algorithmen und Software ausgetauscht und veröffentlicht werden. Eine Plattform für den Austausch bietet das Robotik-Software-Framework ROS, welches als Open-Source-Projekt 2007 ins Leben gerufen wurde und seither durch unzählige Software-Erweiterungen der weltweiten Robotik-Gemeinde wächst. Es bietet neben vielen Sensortreibern, Algorithmen und Werkzeugen eine grundlegende Systemarchitektur und eine Kommunikationsinfrastruktur. Ein solches

Robotik-Software-Framework bietet zudem viele Vorteile für die Entwicklung von interaktiven Versuchsständen. Soweit dem Autor bekannt, war das in dieser Arbeit entwickelte System das Erste, in dem das Robotik-Software-Framework ROS im Bereich der virtuellen Verhaltensforschung eingesetzt wurde.
In diesem Kapitel wird zum einen beschrieben, was bei der Auswahl von Hardwarekomponenten für interaktive Versuchsstände zu beachten ist (2.2). Zum anderen wird gezeigt, in welcher Form ein Robotik-Software-Framework als Basis der Systemarchitektur für Versuchsstände genutzt werden kann und wie dieses in der Praxis umgesetzt wurde (2.3). Im folgenden Unterkapitel 2.1 werden zunächst die Anforderungen an einen interaktiven Versuchsstand definiert.

2.1 Anforderungsprofil

Die Basis des Versuchsstandes für interaktive, virtuelle Fische liegt in der möglichst naturgetreuen virtuellen Nachbildung von realen Fischen. Diese umfasst die äußere Erscheinung der Fische einschließlich deren Bewegung. Im Gegensatz zum realen Fisch bieten virtuelle Fische in der Verhaltensforschung den Vorteil, dass ihr äußeres Erscheinungsbild, ihr Bewegungsmuster und auch ihr → Verhalten (Biologie) beliebig angepasst werden können. Des Weiteren spielt ein solches Forschungssystem besonders unter Versuchsbedingungen seine Vorteile aus: Das Verhalten ist im Gegensatz zu Versuchen mit realen Fisch-Stimuli in jedem neuen Versuchsdurchlauf reproduzierbar und Forschungsfragen lassen sich dadurch gezielter untersuchen. Da das Haupteinsatzgebiet des Systems die verhaltensbiologische Forschung ist, muss davon ausgegangen werden, dass die späteren Benutzer in Hinblick auf die informationstechnischen Grundlagen als fachfremd einzustufen sind. Aus diesem Grund ist ein **hoher Automatisierungsgrad** und eine **hohe Bedienerfreundlichkeit** anzustreben. Des Weiteren wird angenommen, dass sich die Forschungsfragen im Laufe der Zeit verändern und der Versuchsstand entsprechend angepasst werden muss. Aus diesem Grund ist bei der Entwicklung des System auf eine **leichte Erweiterung** und eine **einfache Umgestaltung** Wert zu legen. Dies gilt zum einen für die verwendete Hardware: Versuchsaufbauten mit einer beliebigen Anzahl an Bildschirmen, die Erweiterung durch Aktorik oder die Verwendung anderer Sensoren zur Umgebungsmodellierung sind denkbar. Zum anderen muss auch

die Möglichkeit bestehen, das System einfach durch neue Spezies und damit einhergehende Fortbewegungs- und Verhaltensmodelle zu erweitern. Um die Auswertung der Versuche zu erleichtern sind die erforderlichen Sensordaten zeitlich hoch aufgelöst und synchronisiert aufzuzeichnen. Darüber hinaus muss die Möglichkeit gegeben werden, diese Sensordaten durch Softwarewerkzeuge auszuwerten oder für externe Forschungssoftware aufzubereiten.

2.2 Hardwareaufbau

Der interaktive Fisch-Versuchsstand besteht aus mehreren Komponenten, die in Echtzeit zusammenarbeiten müssen. Zu diesen zählen das Kamerasystem, welches die realen Fische in Echtzeit verfolgt (→ Tracking (Video)), das Animationssystem, welches die virtuellen Fische in Echtzeit erzeugt (→ rendern) und auf visuellen Ausgabegeräte darstellt, ein Kommunikationssystem, welches die Kommunikation zwischen den einzelnen Komponenten in Echtzeit ermöglicht und eine Bedienerschnittstelle zur manuellen Steuerung von virtuellen Fischen (vgl. Abbildungen 2.2 und 2.1). Ein besonderes Augenmerk muss bei der Hardware-Auswahl auf das Kamerasystem und das visuelle Ausgabesystem gelegt werden, da diese je nach Fischart verschiedene Anforderungen erfüllen müssen. Im Folgenden werden die Kriterien zur Auswahl genauer erläutert.

2.2.1 Kamerasystem

Ziel des späteren Systems ist es die → Pose (Robotik) der Fische zuverlässig im dreidimensionalen Raum (*3D*) zu tracken und mit Hilfe des Analyse-durch-Synthese Verfahrens die geometrischen Eigenschaften des realen Fisches in Modellparameter des virtuellen Modells zu übertragen. Im Bereich der 3D-Bewegungsverfolgung beim Menschen haben besonders RGB-D-Kameras, wie der von Microsoft vertriebene Sensor *Kinect* die Forschung massiv vorangetrieben (vgl. [155]). Diese Art von Sensoren bauen neben einem RGB-Farbbild (→ RGB) ein Tiefenbild auf, indem ein Lichtmuster auf die Szene projiziert und dieses Muster vom Kamerasensor aufgenommen wird. Da die Kamera und der Projektor in einiger Entfernung zueinander auf dem Sensor angebracht sind, lässt sich nach der Zuordnung von Muster(Projektor) zu Pixel (Kamera) die Entfernung zum reflektierenden Objekt durch Triangulation bestimmen

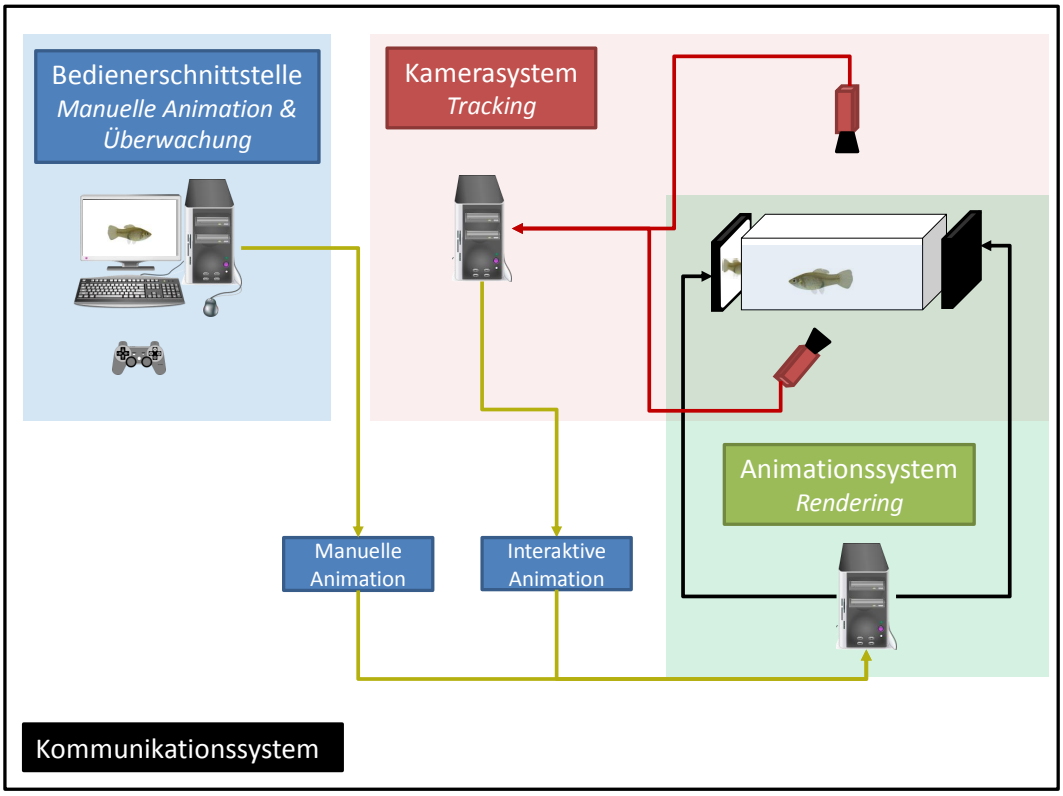


Abbildung 2.1: **Schematischer Aufbau des Versuchsstandes.** Das Gesamtsystem ist in der Darstellung in die vier Komponenten Bedienerschnittstelle, Kamerasystem, Animationssystem und Kommunikationssystem unterteilt.

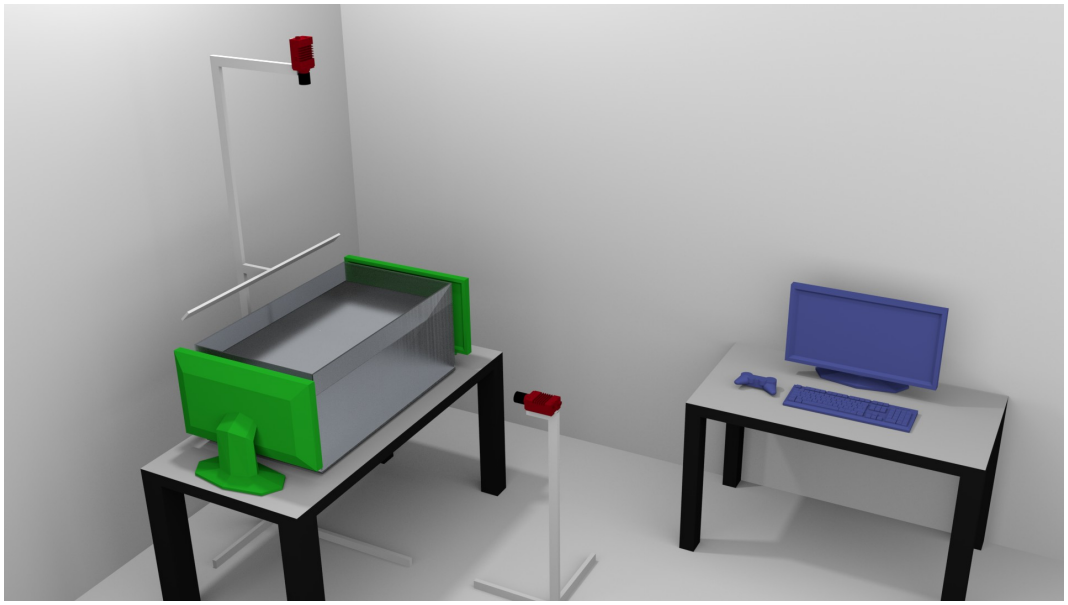


Abbildung 2.2: **Aufbau des Versuchsstandes.** Auf der linken und rechten Seite des Aquariums sind zwei Bildschirme (grün) platziert, auf denen die virtuellen Fische dem Versuchsfisch im Aquarium präsentiert werden. Es ist je eine Kamera über und vor dem Becken angebracht (rot), welche die Bewegung des realen Fisches aufnehmen. Der Experimentator überwacht den Versuch vom Bedienerstand (blau) aus und kann aktiv eingreifen.

(vgl. [133, S.383 ff.]). Wird dieser Schritt für jedes Pixel wiederholt erhält man ein Tiefenbild. Auch in diesem Projekt wurde der Einsatz dieses Sensors zum 3D-Fisch-Tracking evaluiert. Während der Tests stellte sich heraus, dass besonders durch die Reflektion des projizierten Musters an der Aquariumsscheibe bzw. an der durchaus auch bewegten Wasseroberfläche der Sensor nur ein sehr lückenhaftes Tiefenbild erzeugt, welches für das aktuelle Projekt nicht geeignet ist.

Um aktive Beleuchtung und spezielle Projektion zu vermeiden kann die Tiefeninformation auch rein passiv aus den ohnehin vorhandenen Bilddaten gewonnen werden. Im Bereich des visuellen 3D-Fisch-Trackings sind in der Vergangenheit verschiedene Sensor-Systeme eingesetzt worden. Neben Ein-Kamera-Systemen, die den Schattenwurf der Fische zur Positionsbestimmung nutzen (z. B. [67]) oder mit Hilfe von Spiegeln mehrere Ansichten in einem Kamerabild bündeln (z. B. [157]), haben sich besonders Multi-Kamera-Aufbauten durchgesetzt (vgl. [36]). Dabei sind die Kameras meist orthogonal zueinander ausgerichtet und um den Fisch herum angeordnet (siehe Kapitel 3.3). Auch für den hier beschriebenen Versuchsstand wurde ein Zwei-Kamera-System gewählt. Eine Kamera ist oberhalb des Beckens angebracht und dient zum einen zur Bestimmung der Positionstiefe. Sie liefert zum anderen Bilder, welche die Körperbiegung der Fische gut abbilden. Die zweite Kamera nimmt die Längsseite des Aquariums auf (siehe Abbildung 2.2).

Besonders für das Analyse-durch-Synthese-System ist eine möglichst hochauflösende Abbildung der Fische wichtig. Dieses kann erreicht werden, indem man das Fischaquarium möglichst klein wählt, um so den Fisch auf einem möglichst großen Bereich des Kamerasensors abzubilden. Aus Sicht der Verhaltensforschung sollte das Becken, in dem sich der Fisch bewegt, jedoch möglichst groß sein. Um dies zu kompensieren kann eine Kamera mit hoher Auflösung verwendet werden. Kameras mit hoher Auflösung bringen jedoch den Nachteil mit sich, dass die Datenrate steigt und durch die endliche Bandbreite des unterliegenden Kommunikationssystems nur eine geringer Bildrate (siehe Abbildung 2.4 b) möglich ist. Dies führt dazu, dass die Fischbewegung zeitlich nur unzureichend abgebildet wird (siehe Abbildung 2.3). Um eine Kompromisslösung zwischen Auflösung, Framerate und Aquariumsgröße zu finden, müssen folgende Kenngrößen festgelegt werden (siehe auch (Abbildung 2.4 a)):

- **Minimal anzunehmende Fischlänge:**
  Die minimale Fischlänge wird benötigt, um die Abbildungsgröße des Fisches auf dem Bildsensor zu berechnen.

- **Größe des Aquariums:**
  Die benötigte Größe des Aquariums sollte in Hinblick auf den späteren Versuch ermittelt werden. Falls für den Versuch keine standardisierte Größe bekannt ist, empfiehlt sich eine Literaturrecherche nach vergleichbaren Versuchsaufbauten.

- **Framerate:**
  Die benötigte Framerate ist von mehreren Faktoren abhängig: Neben der Bewegungsgeschwindigkeit des Fisches ist entscheidend, welche Informationen aus dem Videomaterial gewonnen werden sollen. Für eine einfache Positionsbestimmung kann eine geringe Framerate ausreichen. Sollen jedoch komplexe Bewegungsabläufe extrahiert werden, muss die Framerate entsprechend höher gewählt werden. Um die minimale Framerate zu ermitteln, empfiehlt sich eine Videoanalyse der Fischbewegung: dazu wird aus allen zu trackenden Körperteilen jenes bestimmt, welches sich am schnellsten bewegt (siehe auch Abbildung 2.3). Nun wird mit einer Kamera mit hoher Framerate eine längere Videosequenz des Fisches aufgenommen. Aus dieser wird eine Szene ermittelt, in der sich das zuvor bestimmte Körperteil sehr schnell bewegt. Durch Zeitmessung der Bewegungsperiode T des Körperteils kann die minimale Framerate ermittelt werden. Im Falle einer Fischschwanzflosse entspricht eine Bewegungsperiode dem Flossenausschlag zu beiden Seiten und endet an der Startposition. Hält man sich an das aus der Signaltheorie bekannte *Nyquist-Shannon-Abtasttheorem* [121] muss die minimale Framerate doppelt so hoch wie der Kehrwert der Periodendauer sein.

- **Länge des Fisches in Pixel:**
  Die benötigte Auflösung des Fisches hängt von der eingesetzten Tracking-Methode ab. Zur einfachen Positionsbestimmung sind nur wenige Pixel erforderlich. Decourt et al. empfehlen eine minimale Fischlänge von 15 Pixeln (vgl. [36]). Sollen jedoch einzelne Körperteile getrackt werden, ist eine deutlich höhere Auflösung notwendig. Im Zweifelsfall ist eine Auflö-

sung zu wählen, bei der alle zu trackenden Körperteile mit dem menschlichen Auge gut zu erkennen sind. Bezogen auf die eingesetzte Kamera und das Aquarium lässt sich die Fischlänge in Pixel (f_p) nach Gleichung 2.1 und Abbildung 2.4 a) berechnen.

$$\begin{aligned} d_1 &= \frac{a}{2tan(\frac{\alpha}{2})} \\ d_2 &= d_1 + b \\ f_p &= \frac{k_x}{2tan(\frac{\alpha}{2})d_2} f \end{aligned} \tag{2.1}$$

- **Bandbreite der unterliegenden Kommunikationsschnittstelle:** Um die Echtzeitfähigkeit des Systems zu gewährleisten, müssen alle Bilddaten in Echtzeit zwischen Kamerasystem und Recheneinheit ausgetauscht werden. Mit zunehmender visueller und zeitlicher Auflösung der Videoaufnahme steigt die benötigte Bandbreite der Kommunikationsschnittstelle (vgl. Gleichung 2.2). Mit steigender Bandbreite nehmen auch die Kosten für das unterliegende Kommunikationssystem zu und können je nach finanziellem Budget die Auswahl des Kamerasystems limitieren. Aus diesem Grund sollte auch die benötigte Bandbreite des Kamerasystems als Kriterium zur Auswahl herangezogen werden. Geht man von einem unkomprimierten Videostream aus, lässt sich die Bandbreite des Kamerasystems aus der Anzahl der Kameras n, Kameraauflösung (k_x, k_y), Bit-Tiefe pro Pixel (k_t), Anzahl der Farbkanäle(k_f) und der Framerate r bestimmen.

$$B[bit/s] = nk_xk_yk_tk_fr \tag{2.2}$$

Für den hier beschriebenen Versuchsaufbau mit Breitflossenkärpflingen wurden die Kenngrößen nach den oben genannten Prinzipien definiert. Als minimale zu erwartende reale Fischgröße wurden 28 mm angenommen. In den durchzuführenden Versuchen zur Partnerwahl werden üblicherweise Aquarien mit einer Größe von 100 x 50 x 40 cm^3 eingesetzt. Durch eine Videoanalyse (siehe Abbildung 2.3) wurde die benötigte Framerate auf 50 Frames/s gesetzt. Besonders für das Analyse-durch-Synthese System ist eine höhere Framerate erstrebenswert, da die Fisch-Modellparameter basierend auf dem vorherigen Frame im nachfolgenden Frame angenähert werden. Sind große Parametersprünge (Fischbewegungen) von Frame zu Frame vorhanden, ist das Risiko

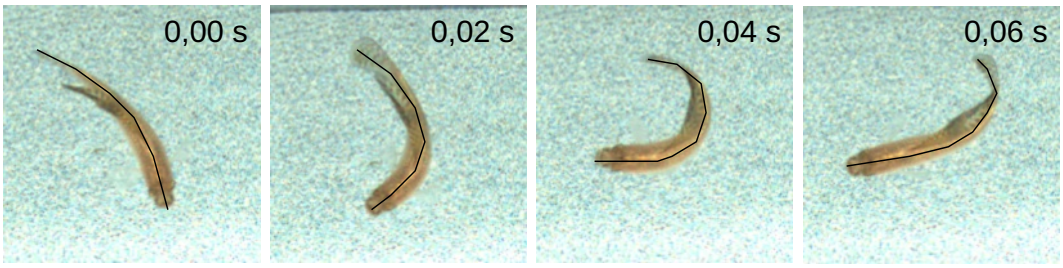


Abbildung 2.3: **Schnelle Richtungsänderung.** Aufnahmen zeigen, dass der Breitflossenkärpfling seine Richtung in 0,06 Sekunden um 90° ändern kann, was einer Winkelgeschwindkeit von ∼26,2 $\frac{rad}{s}$ entspricht.

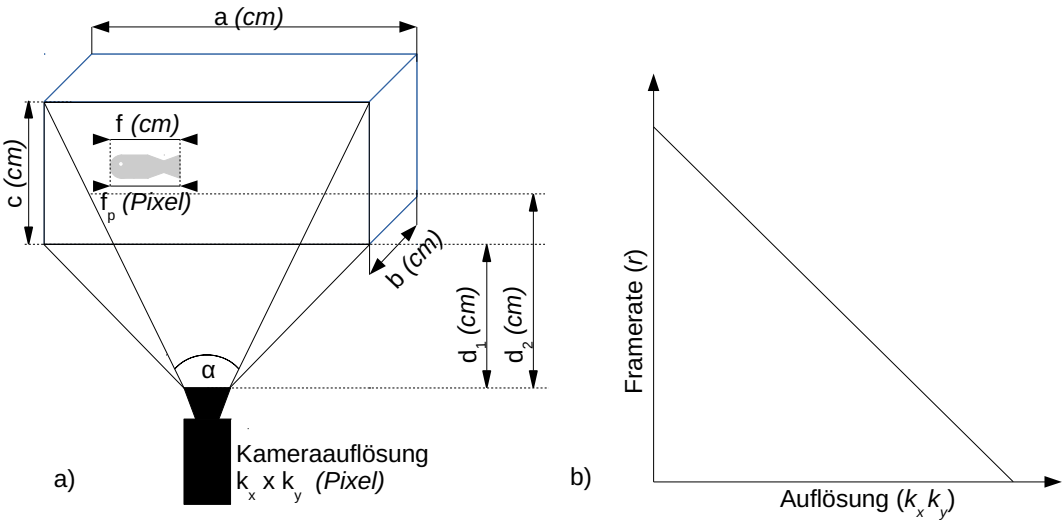


Abbildung 2.4:
a) **Skizze des Kameraaufbaus.** a, b und c geben die Größe des Aquariums an. α beschreibt den Öffnungswinkel der Kamera, d_1 und d_2 geben den Abstand zur vorderen und hinteren Scheibe an. Die Größe des Fisches wird mit f und die minimale Größe im Pixelmaß in f_p angegeben.
b) **Verhältnis zwischen Framerate und Auflösung bei gleichbleibender Datenrate.** Geht man von einer limitierten Bandbreite aus ist das Verhältnis von Framerate r und Auflösung $k_x k_y$ des Videostreams antiproportional zueinander.

der falschen Annäherung erhöht. Für das Analyse-durch-Synthese-System soll hauptsächlich die Position, die Ausrichtung und die Körperbiegung der Fische rekonstruiert werden. Nach Analyse einzelner Fischbilder mit verschiedenen Auflösungen wurde eine minimale Länge von 40 Pixel pro Fisch als ausreichend angesehen. Basierend auf diesem Parameter fiel die Auswahl auf zwei *Prosilica GT 1910c* (Allied Vision Technologies GmbH, Germany) Farbkameras mit einer Auflösung von 1920 x 1080 Pixel, 57 Frames/s und Gigabit-Ethernet-Anschluss.

2.2.2 Visuelles Ausgabesystem

Zu den am häufigsten verwendeten visuellen Ausgabegeräten zählen → Flüssigkristallanzeigen (engl. Liquid Crystal Displays) (LCDs), Videoprojektoren und Röhrenmonitore. Die meisten auf dem Markt verfügbaren Geräte sind für das menschliche Auge optimiert, dessen Wahrnehmungsfähigkeit sich stark von der visuellen Wahrnehmung eines Tieres unterscheiden kann. Das menschliche Auge verfügt über drei verschiedene Arten von Sehzapfen, die als Farbrezeptoren dienen und die Farben rot, grün und blau (RGB) erkennen können. Grundsätzlich werden Lebewesen mit drei verschiedenen Arten an Sehzapfen *Trichromaten* genannt. Tiere haben meist zwischen zwei (*Dichromaten*) und vier (*Tetrachromaten*) unterschiedliche Zapfentypen, können jedoch wie im Fall des Fangschreckenkrebses (*Stomatopoda*) auch zwölf verschiedene Arten von Zapfen haben [139]. Von diesen können sowohl der Wellenlängenbereich des für den Menschen sichtbaren Lichtes als auch die nicht-sichtbaren Bereiche ($<$400 nm [ultraviolett] und $>$700 nm [infrarot]) detektiert werden. Visuelle Anzeigesysteme außerhalb der menschlichen, visuellen Wahrnehmung können nicht oder nur unter sehr hohem finanziellen Aufwand beschafft werden. Im Folgenden werden Bedingungen und Einstellungen aufgezeigt, die eine Nutzung handelsüblicher Anzeigegeräte ermöglichen.

- **Farbdarstellung:**
  Die korrekte Farbdarstellung passend zur visuellen Wahrnehmung der Testtierart ist je nach Forschungsfrage sehr wichtig und kann zum Ausschlusskriterium des Einsatzes von virtuellen Stimuli werden. Um die Parameter der visuellen Wahrnehmung der Testtierart herauszufinden,

empfiehlt sich eine Literaturrecherche. Dabei können auch Parameter von verwandten Arten Aufschlüsse über die Testtierart geben (vgl. [30]). Diese sollten jedoch nur als Richtwert dienen. Ist der wahrnehmbare Wellenlängenbereich und die Anzahl der Sehzapfenarten des Testtieres bekannt, stellt sich die Frage, wie man diesen Bereich mit üblichen Anzeigegeräten abbilden kann. Tedore und Johnsen haben für diesen Zweck eine Umrechnungsformel veröffentlicht, mit der Farben aus dem Farbraum von Dichromaten, Trichromaten und Tetrachromaten in den RGB-Farbraum umzurechnen und um Farben entsprechend des Wahrnehmungssystem möglichst naturgetreu mit RGB-Anzeigegeräten darzustellen [137]. Sollten keine Informationen über den für das Tier sichtbaren Wahrnehmungsbereich bekannt sein, empfehlen Chouinard-Thuly et al. [30] die Farben so einzustellen, dass diese für das menschliche Auge möglichst natürlich aussehen. Bei handelsüblichen RGB-Anzeigegeräten kann eine Ausstrahlung von elektromagnetischer Strahlung außerhalb des sichtbaren Wellenlängenbereichs (unter 400 nm bzw. über 700 nm) nicht ausgeschlossen werden. Dies führt nicht zwangsweise zum Ausschluss von Tierarten, die diesen Wellenlängenbereich detektieren können, sollte jedoch bei der späteren Auswertung berücksichtigt werden.

- **Polarisation:**
  Einige Tierarten können im Gegensatz zum Menschen auch die Polarisation des Lichtes wahrnehmen (vgl. [147]) oder tragen Polarisationsmuster auf dem Körper, wie beispielsweise der Gewöhnliche Tintenfisch (*Sepia officinalis*), die möglicherweise zur innerartlichen Kommunikation dienen (vgl. [122]). Sollte die Testtierart sensitiv für die Polarisation des Lichtes sein, ist dies bei der Auswahl des Anzeigesystems zu berücksichtigen. Besonders → LCDs polarisieren systembedingt das ausgestrahlte Licht. Alternativ können Plasmabildschirme, Röhrenmonitore oder Projektoren, die nicht auf LCD-Technik basieren, eingesetzt werden, da diese das Licht nur im geringen Maße polarisieren.

- **Zeitliche Auflösung:**
  Die visuelle Wahrnehmung von Lebewesen ist zeitlich gesehen träge. Werden Bilder in sehr schneller Abfolge hintereinander gezeigt, wird aus den

statischen, aufeinanderfolgenden Bildern eine flüssige Bewegung. Je nach Tierart ist die Höhe der Bildrate (engl. *frames per second [fps]*) für einen flüssigen Bildeindruck unterschiedlich. Dies ist auch bei der Auswahl eines Anzeigegerätes zu beachten. Gängige RGB-Anzeigegeräte sind für das menschliche Auge optimiert. Die für einen flüssigen Bildeindruck benötigt Bildrate kann jedoch je nach Tierart stark variieren. Dies führt dazu, dass ein für das menschliche Auge flüssiger Bildeindruck (ab ca. 25 bis 30 Hz), von einigen Tierarten als Abfolge von Einzelbildern wahrgenommen wird und der Eindruck einer Bewegung nicht vermittelt werden kann. Für nicht fliegende Tiere empfiehlt Fleishman und Elders eine Bildrate von 60 Hz oder höher zu verwenden [46]. Des Weiteren gilt es bei älteren Kathodenstrahlröhrenbildschirmen auf die Bildwiederholungsrate zu achten. Anders als bei → LCDs, bei denen der Bildschirm durchgehend belichtet wird, werden die einzelnen Bildpunkte bei Röhrenmonitoren in schneller Abfolge vom Kathodenstrahl zum Leuchten gebracht. Die Leuchtkraft des Bildpunktes ist somit nicht konstant und kann je nach Art als Flimmern wahrgenommen werden. Auch das in der klassischen Fernseh- und Videotechnik eingesetzte Halbbildverfahren, bei dem pro Durchlauf nur jede zweite Zeile des Bildes aktualisiert wird, kann zu Flimmern führen.

- **Räumliche Auflösung:**
  Für den Einsatz virtueller Stimuli sind besonders drei Merkmale, welche die räumliche Auflösung eines Anzeigegerätes beschreiben, wichtig: Bildschirmauflösung (Breite x Höhe in Pixel), Pixeldichte (z. B. Pixel pro Zoll [ppi]) und Pixelabstand (in Millimetern) (vgl. [30]). Zur Auswahl des Anzeigegerätes sollte neben der Sehschärfe der Testtierart auch der spätere Abstand des Testtieres zum Anzeigegerät bekannt sein. Chouinard-Thuly et al. [30] empfehlen grundsätzlich für Versuche, in denen die Testtierart sehr nah an das Anzeigegerät herankommt oder eine hohe Sehschärfe besitzt, ein Anzeigegerät mit sehr hoher Pixeldichte und kleinstmöglichem Pixelabstand zu wählen, um zu verhindern, dass die Tiere einzelne Pixel erkennen können. Eine genauere Berechnung des minimalen Abstandes zwischen Anzeigegerät und Testtier in Bezug zur räumlichen Auflösung ist in der Arbeit von Fleishman und Elders zu finden [46].

Die hier gegebenen Auswahlkriterien sind als Richtwert zu sehen. Vor Versuchsbeginn sollte jedoch eine ausgiebige Validierung erfolgen. Chouinard-Thuly et al. geben dazu in ihrer Arbeit genaue Anweisungen [30].

Für die hier durchgeführten Versuche mit Breitflossenkärpflingen wurden zwei verschiedene Anzeigegeräte getestet: ein 24 Zoll →LCD (EIZO Foris FX2431, EIZO Nanao AG, Österreich, 1920 x 1200 Pixel und ein 19 Zoll Röhrenbildschirm (Samsung SyncMaster 997 MB, Samsung Electronics Display (M) (HSD), Malaysia, 85 Hz, 1280 x 960 Pixel Auflösung). Beide Geräte erfüllen die von Fleishman und Elders [46] geforderten Merkmale bezüglich zeitlicher und räumlicher Auflösung. In einem Vergleichstest wurde gezeigt, dass die Reaktion von Testfischen auf Stimuli, die auf einem LCD präsentiert wurden (Animation und Video), sich nicht von der Reaktion auf lebende Testfische unterschieden. Zudem wurde gezeigt, dass sich Testfische länger vor →LCDs als vor Röhrenmonitoren aufhalten (vgl. [49]). Folglich wird das genannte LCD System verwendet.

2.3 Software

Ein interaktiver Versuchsaufbau für Fisch-Computer-Interaktionen weist viele Ähnlichkeiten zu Robotersystemen auf. Anders als in der Robotik existieren für das Feld der interaktiven Tier-Computer-Systeme zum Zeitpunkt der Erstellung dieser Arbeit keine dem Autor bekannten Software-Frameworks, welche die Software-Entwicklung vereinfachen und standardisieren. Aus diesem Grund wird für den hier beschriebenen Versuchsstand auf ein Software-Framework der Robotik zurückgegriffen. Dies bietet zum einen den Vorteil auf vorhandene Softwaremodule zurückgreifen zu können und damit die Entwicklung zu beschleunigen und zum anderen standardisierte Schnittstellen zu nutzen, womit das Gesamtsystem wie in Unterkapitel 2.1 gefordert leicht verändert und erweitert werden kann. Dies eröffnet die Möglichkeit, den vorhanden Versuchsstand einfach an andere Forschungsfragen anzupassen. Im Folgenden werden weitere Forschungsprojekte vorgestellt, die zum einen Roboter in der Fisch-Verhaltensforschung einsetzten und zum anderen Systeme, die ebenfalls Robotik-Software-Frameworks zur Entwicklung von Versuchsständen eingesetzt haben (siehe Unterkapitel 2.3.1). In Unterkapitel 2.3.2 wird

das Robotik-Software-Framework ROS, welches in diesem Projekt verwendet wird und schon in mehreren Projekten des Institut für Echtzeit Lernsystem erfolgreich eingesetzt wurde (siehe [83, 84]), vorgestellt. In Unterkapitel 2.3.3 wird am Beispiel des hier beschriebenen Systems gezeigt, wie sich interaktive Versuchsstände in das Robotik-Software-Framework ROS implementieren lassen.

2.3.1 Robotiksysteme in der Fisch-Verhaltensforschung

Die Robotik findet schon seit einiger Zeit Anwendung in der Fisch-Verhaltensforschung. Faria et al. entwickelten einen Roboter-Fisch, der auf einem programmierten Pfad auf dem Beckenboden durch das Aquarium bewegt werden konnte [43]. Butail et al. gingen einen Schritt weiter und entwickelten einen ferngesteuerten, freischwimmenden Roboter-Fisch, der mit Hilfe eines visuellen Tracking-Verfahrens auf Kreisbahnen durch das Becken gesteuert wurde. Dabei konnten zum einen die Geschwindigkeit des Roboters und zum anderen die Frequenz des Schwanzflossenschlages verändert werden. Ziel der Arbeit war es, den Einfluss der Fortbewegung eines Zebrabärblings (*Danio rerio*) auf einen Schwarm derselben Art zu untersuchen [18]. Ähnlich wie Faria et al. entwickelte Landgraf et al. einen Fisch-Stimulus der computergesteuert mit Magneten in definierter Höhe über dem Boden des Aquariums bewegt werden konnte. Im Gegensatz zu Faria et al. konnte der Fisch-Stimulus bei Landgraf et al. zur Laufzeit beliebig durch das Becken gesteuert werden. Zusätzlich wurde sowohl der Roboter als auch der reale Guppy-Fischschwarm (*Poecilia reticulata*) im Becken durch ein Bildverarbeitungssystem getrackt und der Fisch-Stimulus konnte auf Basis dessen aktiv auf das Verhalten des Schwarms reagieren [66]. In einer späteren Arbeit wurde die Akzeptanz des Roboterfisches durch naturgetreue Augen und eine natürlichere Fortbewegung erhöht [65].

Neben dem direkten Einsatz von Robotiksystemen in der Verhaltensforschung werden auch Teilkomponenten und Systemarchitekturen aus der Robotik für die Verhaltensforschung bei Fischen genutzt. Robotik-Software-Frameworks bündeln viele Teilkomponenten, Algorithmen, Sensortreiber und Werkzeuge der Robotik und haben sich in den vergangen Jahren gegen proprietäre Software mehr und mehr durchgesetzt. Eins der bekanntesten Robotik-

Software-Frameworks ist das *Robot Operating System (ROS)*[1]. Dieses Open Source →Framework zeichnet sich besonders durch eine große und vielseitige Entwicklergemeinde aus (siehe auch Kapitel 2.3.2) und wurde auch für das in dieser Dissertation erarbeitete System zur Verhaltensforschung mittels interaktiver, virtueller Stimuli genutzt. Im Folgenden werden weitere Projekte vorgestellt, in denen ebenfalls Robotik-Frameworks zur Verhaltensforschung bei Fischen eingesetzt wurden. Im Jahr 2015 wurde das *FENOMENO*[2] Projekt ins Leben gerufen. Im Projekt wurden die Auswirkungen von Nanopartikeln auf das Ökosystem untersucht. Um deren Auswirkungen auf Kleinstlebewesen zu untersuchen, wurde überprüft, ob Wasserflöhe (*Daphnia magna* und *Daphnia pulex*) und Zebrabärblinglarven (*Danio rerio*) unter Zugabe von Nanopartikeln Verhaltensänderungen aufweisen. Um diese zu detektieren, wurden die Bewegungen der Kleinstlebenwesen mit Hilfe von Kameras aufgenommen und diese in Nachbereitung aus den Videos extrahiert. Dabei wurde eine einzelne oder wurden zwei orthogonal zueinander ausgerichtete Infrarot-Kameras eingesetzt. ROS wurde als →Grafische Benutzeroberfläche eingesetzt und lieferte Treiber für die Kameras, ein Werkzeug zum Kalibrieren und die Möglichkeit die Aufnahmen in Echtzeit und synchronisiert für die spätere Auswertung zu speichern [63].
Stowers et al. bauten einen Virtual Reality Experimentierstand für Insekten und Spinnen auf (*FlyVR*[3]). Das entwickelte System diente der neurobiologischen Forschung und wurde eingesetzt, um die optische Wahrnehmung einer Taufliegen Art (*Drosophila*) zu erforschen. Dazu wurde die Fliege in einen abgeschlossenen Zylinder gesetzt, auf dessen Wänden interaktive 3D-Animationen projiziert wurden. Die Projektion wurde in Echtzeit auf die aktuelle Postion der Fliege abgestimmt, um einen möglichst realistischen 3D-Effekt der virtuellen Welt zu erzeugen. Dazu wurde die Fliege mit Hilfe mehrerer Kameras getrackt. Die so gewonnenen Positionsdaten wurden anschließend dem Animationssystem übergeben und die Projektion entsprechend der Fliegenposition verändert. Das System wurde auch mit einer Spinne (*Cupiennius salei*) getestet [131]. Aufbauend auf diesem System entwickelten Stowers et al. das

[1]Robot Operating System. `http://www.ros.org`
[2]https://www.uni-siegen.de/fokos/forschungsprojekte/fenomeno/?lang=de
[3]www.flyvr.org

Virtual Reality Framework *FreemoVR*[4], welches auch für Mäuse und Zebrafische (*Danio rerio*) eingesetzt werden kann. Im Fall von Mäusen wurde eine Bodenprojektion gewählt. Oberhalb der Projektion ist ein Ring befestigt, auf dem sich die Maus frei bewegen kann und dabei über die Kante des Rings die Projektion der virtuellen Welt sieht. Dabei wird die Position des Mäusekopfes mit Hilfe eines Bildverarbeitungssystems getrackt und die Projektion der virtuellen Welt auf die Position der Maus angepasst. Für die Virtual Reality Versuchsanordnung bei Zebrafischen wurde eine mit Wasser gefüllte Halbkugel gewählt. Von der Unterseite wurde die virtuelle Welt auf die Halbkugel projiziert. Das System ist für einen einzelnen Fisch ausgelegt, der ebenfalls mit Kameras, die oberhalb der Schale angeordnet sind, beobachtet und durch ein Bildverarbeitungssystem getrackt wird [132]. Sowohl *FreemoVR* als auch *FlyVR* basieren auf dem Framework ROS.

2.3.2 Robotik-Software-Framework ROS

Neben einigen anderen Robotik-Software-Frameworks hat sich besonders die freie Open-Source-Software *Robot Operating System ROS*[5] durchgesetzt. Mittlerweile umfasst das Framework mehr als 3000 Software-Pakete, wie z.B. Hardware-Treiber, Algorithmen und Visualisierung-Werkzeuge. Eine besondere Stärke der Software ist jedoch die aktive Entwickler-Gemeinschaft, die in Foren (mehr als 5700 Benutzer), im online Kompendium (mehr als 3300 Benutzer), in Mailinglisten (mehr als 1500 Teilnehmer) und auf Online-Plattformen Hilfe und Unterstützung für ROS-Entwickler anbietet (vgl. [110]). Zur Philosophie des Robotik-Frameworks gehört es, die Software-Komplexität niedrig zu halten, auch wenn dies unter Umständen auf Kosten der Effektivität geht. Um dieses Ziel zu erreichen, setzen die Entwickler von ROS auf das Micro-Kernel Design: im Gegensatz zur monolitischen Software-Architektur, in der alle Komponenten in einer untrennbaren und homogenen Struktur verbunden sind, wird der Systemkern der Micro-Kernel Architektur nur für sehr grundlegende Aufgaben genutzt und fast alle Funktionen und Komponenten ausgegliedert. ROS kann somit als verteiltes System angesehen werden. Die Komponenten des Systems kommunizieren direkt miteinander (Peer-to-Peer). Ein zentraler Server

[4] https://strawlab.org/freemovr-press

[5] Robot Operating System. `http://www.ros.org`

wird nur zur Bekanntmachung der beiden Kommunikationspartner zu Beginn der Verbindungen benötigt. Diese Architektur hat neben der Reduktion der Komplexität zudem den Vorteil, dass Prozesse, die große Mengen von Daten austauschen, nicht einen zentralen Kontenpunkt belasten. Des Weiteren bietet dieses Architektur-Modell den Vorteil, dass einzelne Komponenten je nach Aufgabenstellung einfach ersetzt oder mit anderen Komponenten kombiniert und für weitere Aufgabenstellungen eingesetzt werden können. Besonders in der Forschung, in der zur Beantwortung einer Forschungsfrage verschiedene Methoden ausprobiert werden müssen, spielt diese Architektur ihre Vorteile aus.

Die zentrale Komponente im ROS-System ist der *ROS Master*. Dort werden alle Komponenten und deren Adressen verwaltet. Im ROS-System werden die einzelnen Komponenten als *ROS nodes* bezeichnet. Diese stellen ein Softwaremodul dar und können z.B. Hardware-Treiber (z.B. für Kameras, Motor-Steuergeräte oder Joysticks), Algorithmen (Lokalisierungs- oder Navigationsalgorithmen) oder grafische Oberflächen abbilden. Die Nodes kommunizieren über *Messages* miteinander. Messages können einfache Datentypen (z.B. String, Float, Integer) oder Kombinationen aus vielen verschiedenen Datentypen beinhalten und können je nach Anwendungsfall konfiguriert werden. Nodes, die Messages verschicken, werden als *Publisher* und Nodes, die Messages empfangen, als *Subscriber* bezeichnet. Damit der Publisher von anderen Nodes gefunden werden kann, registiert sich dieser beim Master unter einem sogenannten *Topic*, der den Namen des Nachrichtenkanals darstellt. Empfangende Subscriber können sich beim Master mit Hilfe des Topics die genaue Adresse des Publishers holen und so eine Peer-to-Peer Verbindung aufbauen.

Messages werden grundsätzlich für einen kontinuierlichen Datenstrom zwischen Nodes genutzt, wie z.B. bei Sensordaten. Zum Verschicken einzelner Befehle an eine Node bieten sich *Services* an. Diese werden z.B. eingesetzt, um einzelne Prozesse in der Node zu starten oder um explizit einen Datensatz abzufragen.

Um Nodes zu konfigurieren und um Konfigurationsparameter zu setzen, bieten sich das *dynamic_ reconfigure*-Paket an. Mit Hilfe dieses Paketes lassen sich zuvor definierte Parameter zur Laufzeit mit Hilfe einer grafischen Oberfläche oder einer textbasierten Eingabe verändern.

Besonders in der Forschung ist das *Logging und Playback*System `rosbag` des

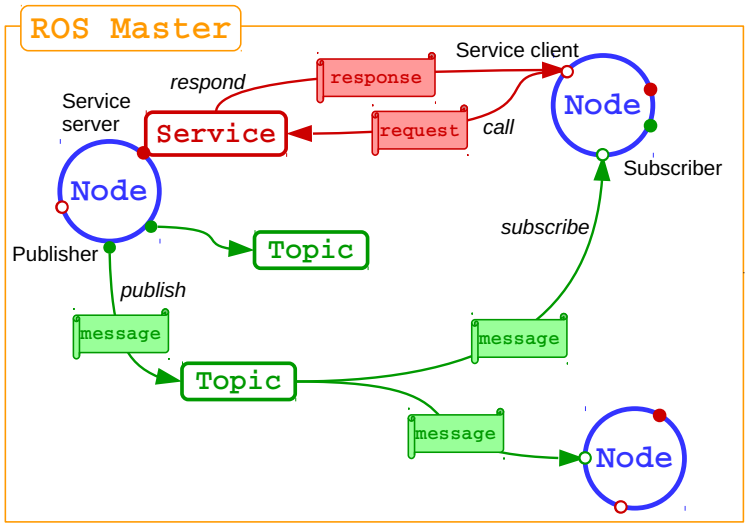


Abbildung 2.5: **ROS Struktur.** Die Abbildung zeigt das Zusammenspiel zwischen Nodes. Die Grafik wurde in Anlehnung an [151] erstellt.

ROS-Systems sehr hilfreich. Dies ermöglicht es, Messages mit Zeitstempeln synchron in sogenannten *BAG*-Dateien aufzunehmen und zu einem späteren Zeitpunkt abzuspielen. Ein möglicher Anwendungsfall wäre die Aufnahme von Kamerabildern, an Hand derer ein Algorithmus zur Fisch-Detektion erprobt werden soll. Während der Entwicklung des Algorithmus kann die Videosequenz beliebig oft abgespielt werden, um so den Algorithmus auf realen Sensordaten zu erproben (vgl. [105]). Eine vereinfachte Übersicht des Systems ist in Abbildung 2.5 zu finden.

2.3.3 Integration des Forschungsaufbaus in das Robotik Framework ROS

Entsprechend des in ROS verwendeten Micro-Kernel-Designs werden auch die Komponenten des Forschungsaufbaus in mehrere Nodes aufgeteilt. Neben der Komplexitätsreduktion bietet dies die Möglichkeit, einzelne Komponenten je nach Leistungsbedarf auf unterschiedliche Rechner zu verteilen. Im hier gegebenen Projekt wurden aus Gründen der benötigten Rechenleistung die Funktionsbereiche Fisch-Tracking, Animation und Posenschätzung auf unterschiedliche Rechner verteilt. Die Kommunikation zwischen diesen Funktionsbereichen ist durch den ROS-Master gewährleistet und wird über → Ethernet umgesetzt.

Im Folgenden werden diese Funktionsbereiche genauer erläutert. Eine vereinfachte Übersicht der System-Struktur ist zudem in Abbildung 2.6 zu finden.

2.3.3.1 Fisch-Tracking

Ausgangspunkt des Funktionsbereiches Tracking sind die beiden `prosilica_camera`-Nodes. Diese sind frei verfügbar und liefern das Rohbild der Farbkamera, welches als → Bayer-Matrix bereitgestellt wird. Um einen Namenskonflikt der beiden Kamera-Topics vorzubeugen, werden die Topic-Namen mit der jeweiligen Lage der Kamera ergänzt (left, right). Zur Umrechnung des Rohbildes in ein Farbbild bietet ROS die Node `image_proc` an. Diese kann zudem zum Entzerren des Kamerabildes genutzt werden. Die dafür benötigten Verzerrungsparameter können ebenfalls durch eine ROS-Node (`camera_calibrator`) während der intrinsischen → Kamerakalibrierung geschätzt werden.
Für die extrinsische → Kamerakalibrierung ist die `fishtank_calibrator`-Node entwickelt worden, die vor Beginn des Versuches auf Basis von Markern die extrinsischen Kameraparameter im Bezug zum Aquarium bestimmt (eine genaue Beschreibung ist im Kapitel 3.4 zu finden). Um sich die konvertierten und entzerrten Bilder in Echtzeit anzeigen zu lassen, bietet sich die von ROS bereitgestellte Node `image_view` an. Zentrale Node des Funktionsbereiches Trackings ist die `fish_detector`-Node. Mit Hilfe verschiedener Verfahren berechnet die Node 3D-Position und Ausrichtung der Fische oder extrahiert die Fisch-Silhouetten aus den Kamerabildern. Eine genaue Beschreibung der Verfahren ist im Unterkapitel 3.8 zu finden. Mit Hilfe der Node `rosbag_record` lassen sich die Kamerabilder mit Zeitstempel persistent abspeichern. Mit Hilfe der Node `rosbag_play` lassen sich diese nachher erneut Abspielen und können so zum Beispiel zur Validierung des Tracking-Ergebnisses genutzt werden. Die gewonnenen Daten wie Position und Ausrichtung sowie Fisch-Silhouetten werden auf den Topics `fish_poses` und `fish_silhouettes` veröffentlicht und können in den nächsten Funktionsbereichen Animation und Posenschätzung in Echtzeit verwendet werden.

2.3.3.2 Animation

Der Funktionsbereich Animation gliedert sich wiederum in die Unterbereiche manuelle, automatische und interaktive Animation. Die genaue Funktionswei-

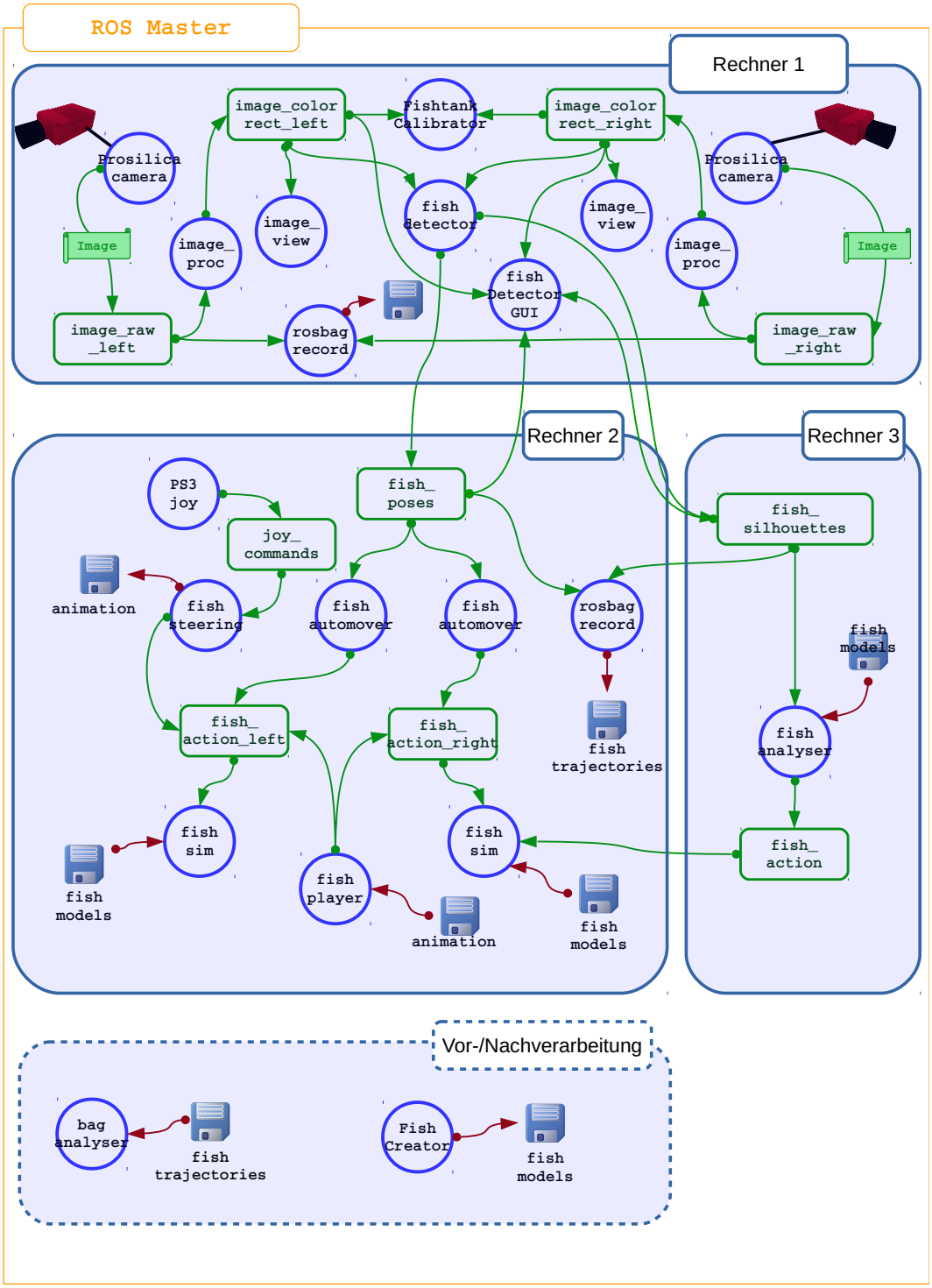


Abbildung 2.6: **Implementierung des Forschungsaufbaus mittels des Robotik-Software-Frameworks ROS.** Die blauen Kreise stellen Nodes, die grünen Boxen Topics dar. Die Diskette symbolisiert ein Lesen von oder Schreiben auf ein Medium.

se der verschiedenen Verfahren wird in Kapitel 4 beschrieben.
Zentrale Node aller Animationsverfahren ist die `fishsim`-Node. Diese stellt auf Basis einer →Spiel-Engine einzelne oder mehrere Fische auf dem Anzeigegerät dar. Dabei kann eine Instanz dieser Node jeweils ein Anzeigegerät bedienen. Grundsätzlich können nahezu beliebig viele dieser Nodes verwendet werden. Die Animationsparameter für die zu animierenden Fische erhält die Node über das Topic `fish_action_{left,right}`. Die Message ist vom Typ `ActionArray` (ein `ActionArray` besteht wiederum aus beliebig vielen `Action`-Messages) und enthält für jedes in `fishsim` geladene Modell die genaue Position, Rotation und Knochenstellung. Die Fisch-Modelle werden aus einem zuvor definierten Ordner geladen. Um zur Laufzeit Optionen in der Node zu ändern oder Informationen zu bekommen, sind mehrere Services implementiert, die sowohl von anderen Nodes als auch vom Benutzer direkt aufgerufen werden können:

- **getModels:** liefert die Namen der aktuell geladenen Fisch-Modelle.
- **getModelBones:** sendet alle Knochennamen zu einem zuvor im Call definerten Modell. Alle `Action`-Messages werden auf Basis des Modell- und Knochennamens in `fishsim` zugeordnet.
- **exchangeModel:** bietet die Möglichkeit vorhandene 3D-Fisch-Modelle zur Laufzeit in `fishsim` auszutauschen, zu entfernen oder neue Modelle hinzuzufügen.
- **scaleFish:** passt die Größe des Fisches während der Animation an.
- **setLight:** fügt Lichtquellen der virtuellen Umgebung hinzu oder ändert vorhandene.

Um Einstellungen wie Fenstergröße und Position der `fishsim`-Anzeige während der Laufzeit über eine Benutzerschnittstelle einstellen zu können, wurde darüber hinaus die `dynamic_reconfigure`-Umgebung implementiert. Zur **manuellen Animation** wir die Node `fish_steering` genutzt. `fish_steering` erhält von der `ps3_joy`-Node Steuerdaten des →Gamecontrollers. Diese Daten werden dann von der Node in Fisch-Bewegungen umgesetzt und in Form

von `ActionArray`-Nachrichten an die `fishsim`-Node gesendet. Um Animationen zu speichern und diese zu einem späteren Zeitpunkt abzuspielen, können die `ActionArray`-Nachrichten auch aufgezeichnet und später durch die `FishPlayer`-Node abgespielt werden.
Die **automatische** und **interaktive Animation** wird durch die `fish_automover`-Node umgesetzt. Diese enthält ein Fisch-Bewegungsmodell und kann den Fisch naturgetreu und zufällig durch das Becken steuern. Wie auch die `fish_steering`-Node sendet auch diese Node `ActionArray`-Nachrichten zur `fishsim`-Node. Um den virtuellen Fisch **interaktiv** zu Steuern und somit direkt auf Bewegungen des echten Fisches zu reagieren, verfügt die Node zudem über die Möglichkeit, die Positions- und Posen-Daten des realen Fisches über das Topic `fish_poses` zu empfangen und den Fisch entsprechend dieser Daten zu steuern. Des Weiteren sind zwei Services in der Node implementiert: `enable_mimic` ermöglicht es, die automatische Bewegungen der Rückenflosse und des → Gonopodiums einzuschalten; `enableAutoMover` dient der Aktivierung dieser Node und damit der automatischen Fischbewegung. Beide Services werden z.B. in der `fishplayer`-Node genutzt. Die `fishplayer`-Node dient der automatischen Animationswiedergabe auf mehreren Anzeigegeräten. Dazu können Playlists definiert werden, die einen Versuchsablauf beschreiben. Sie bietet zudem die Möglichkeit, die automatische Fischanimation (`fish_automover`) zeitgesteuert zu starten oder zu beenden.

2.3.3.3 Posenschätzung

Im Funktionsbereich der Posenschätzung wird auf Basis der Fischsilhouetten (Node `fish_detector`, Topic `fish_silhouettes`) die 3D-Pose und die Bewegung des Fisches geschätzt. Die `fish_analyser`-Node nutzt dazu einen Optimierungsalgorithmus, der die reale Fisch-Silhouette mit einer synthetisch erzeugten Silhouette vergleicht, um so die genauen Fischmodel-Parameter zu schätzen. Der Algorithmus nutzt dieselben Fisch-Modelle wie die `fishsim`-Node. Die geschätzten Modell-Parameter können als `ActionArray`-Nachricht an die `fishsim`-Node versendet und dort dargestellt werden.

2.3.3.4 Nodes der Vor- und Nachverarbeitung

Neben den Nodes, die in Echtzeit miteinander kommunizieren, sind einige Prozesse zeitlich nicht kritisch und können vor- oder nachgelagert zur Prozesskette ausgeführt werden. In machen Fällen bietet sich trotzdem eine Implementierung als ROS-Node an, um Schnittstellen des Frameworks zu nutzen. In diesem Projekt wurde in vor- und nachgelagerten Prozessen sowohl Software mit als auch ohne ROS-Schnittstelle verwendet. Im Fall des `FishCreator`, welcher der Gestaltung des Fisch-Stimulus und der Generierung des 3D-Modells dient, wurde auf die Implementierung von ROS-Schnittstellen verzichtet. Alle Modelle werden in einem gängigen 3D-Modell-Format (*DirectX*) abgespeichert und können dann durch die Simulations-Nodes direkt vom unterliegenden Speichermedium geladen werden. Da keine weiteren Informationen übertragen werden müssen, wurde hier auf eine ROS-Schnittstelle verzichtet.
Die `bag_analyser`-Node hingegen wurde in der ROS-Umgebung implementiert und dient der Auswertung von Tracking-Daten nach einem Versuch. Mit Hilfe der Software kann der Experimentator aufgenommene Positions- und Posendaten (Topic `fish_poses`) analysieren und so für die Forschungsfrage relevante Daten extrahieren. Die Daten werden während des Versuches als Bag (Bezeichnung der `rosbag`-Datei, die auf dem Speichermedium abgelegt wird und die aufgenommenen Daten erhält) aufgezeichnet und können anschließend mit Hilfe der Software eingelesen werden. Grundsätzlich bietet es sich an ROS zur Datenaufzeichnung während eines Versuches zu nutzen. Alle Daten werden mit präzisem Zeitstempel (im Nanosekundenbereich) aufgenommen und können zu einem späteren Zeitpunkt eingelesen oder abgespielt werden. Es können beliebig viele Topics in einem Bag zusammengeführt werden, was besonders in Versuchen mit vielen Sensordaten von Vorteil ist.

2.4 Fazit

In diesem Kapitel wurden zum einen die Hardwareauswahl und zum anderen die Systemarchitektur des Versuchstandes auf Basis eines Robotik-Software-Frameworks erläutert. Die Hardwareauswahl beschränkte sich auf die Komponenten, die für den speziellen Einsatz mit Fischen entscheidend sind: die Kamera und das Anzeigegerät. Zur Auswahl der passenden Kamera wurden

mehrere Kriterien definiert, welche unter anderem auf artspezifischen Eigenschaften basieren.
Bei der Auswahl des richtigen Anzeigegerätes ist besonders auf die artspezifische visuelle Wahrnehmung zu achten und vor Versuchsbeginn eine Validierung der visuellen Wahrnehmung des verwendeten Anzeigegerätes durchzuführen. Um bei der Entwicklung des interaktiven Versuchsstandes auf vorhandene Komponenten zurückgreifen zu können, die Entwicklung zu standardisieren und ein hohes Maß an Flexibilität in Hinblick auf Erweiterbarkeit zu gewährleisten, wurde ein Robotik-Software-Framework als Basis der Systemarchitektur verwendet. Das Open-Source-Framework ROS wurde in diesem Zusammenhang vorgestellt und die praktische Umsetzung des interaktiven Versuchsstandes auf Basis dieses Frameworks gezeigt.
Es wurde im Rahmen dieser Dissertation – soweit dem Autor bekannt – erstmals (Bezugszeitpunkt: Veröffentlichung der Publikation [87]) gezeigt, dass sich ROS als sehr nützliches Werkzeug erweist, um computergestützte Verhaltensforschung zu standardisieren. Besonders durch die Open-Source-Politik entstehen den Benutzern keine weiteren Kosten und es können alle Komponenten nach Belieben verändert oder das Projekt durch neue Komponenten erweitert werden. Zudem stellt ROS durch das BAG-Konzept eine gute Möglichkeit dar, Sensordaten während der Versuche aufzunehmen und diese der Forschungsgemeinschaft zur Verfügung zu stellen. Im Rahmen des durch die Deutsche Forschungsgemeinschaft geförderten Projektes mit dem Titel „Analyse durch Synthese mit virtuellen Fischen als neue Versuchsmethode in Untersuchungen zur Partnerwahl“, das in Kooperation zwischen dem Institut für Echtzeit Lernsysteme und dem Institut für Biologie (KU 689/11-1 und Wi 1531/12-1) an der Universität Siegen durchgeführt wurde und die Grundlage für diese Dissertation bildet, wurde ein erster Schritt in diese Richtung unternommen. Das ROS-basierte Animations- und Präsentationswerkzeug für virtuelle Fisch-Stimuli *FishSim Animation Toolchain*[6] wurde als Open Source Projekt veröffentlicht und steht nun der internationalen Forschungsgemeinschaft zur Verfügung.

[6]https://bitbucket.org/EZLS/fish_animation_toolchain/wiki/Home

Kapitel 3

Echtzeit Fisch-Tracking unter Berücksichtigung der Lichtbrechung

Das Tracken von Fischen ist schon längere Zeit Gegenstand der Forschung und hat Anwendung in vielen Bereichen gefunden. Neben der Fischerei [124] finden Tracking-Systeme insbesondere Anwendung in der Fisch-Verhaltensforschung. Dabei werden sowohl Fische und Fischschwärme in offenen Gewässern (z.B. [1, 16, 37, 45]) als auch in Becken oder Aquarien getrackt (siehe Kapitel 3.3). Für größere Fische in offenen Gewässern werden Funksender (z.B. [1]) oder akustische Sender eingesetzt, die dem Fisch angehängt (z.B. [16]) oder implantiert (z.B. [37]) werden. Auf Grund der begrenzten Sicht unter Wasser werden Video-Tracking-Systeme in offenen Gewässern nur zur Überwachung von kleinen Bereichen wie Korallenriffe (vgl. [45]) eingesetzt. In kleinen Becken und Aquarien werden hingegen mehrheitlich Video-Tracking-Systeme verwendet, die auch Gegenstand dieser Arbeit sind und im Folgenden betrachtet werden. Im Hinblick auf die Anwendung im Versuchsstand mit interaktiven virtuellen Fischen muss das Tracking-System zudem weitere Voraussetzungen erfüllen, die in Unterkapitel 3.2 erläutert werden. In Unterkapitel 3.1 werden zum besseren Verständnis dieses Kapitels Begriffe eingeführt und grundlegende Techniken erläutert. Bisherige Arbeiten auf dem Gebiet des Fisch-Trackings werden in Unterkapitel 3.3 vorgestellt. Zum Tracken der Fische in 3D ist eine präzise Kalibrierung der Kameras vonnöten. Das speziell auf diesen Anwen-

dungszweck angepasste Verfahren wird in Kapitel 3.4 vorgestellt. Des Weiteren bringt die Lichtbrechung, verursacht durch den Wechsel der Medien (Wasser, Glas und Luft), besondere Schwierigkeiten mit sich. Das zur Kompensation der Lichtbrechung entwickelte Verfahren wird in Unterkapitel 3.5 vorgestellt. In Unterkapitel 3.6 werden die eigentliche Detektion der Fische und die mit dem Tracking in Aquarien verbundenen Probleme behandelt. In dieser Arbeit wurde zudem ein Verfahren entwickelt, welches zwei Konturen eines Objektes aus unterschiedlichen Perspektiven zu einem einfachen 3D-Modell umwandeln kann (Unterkapitel 3.7). Die nach der Detektion anschließende Verfolgung der Fische im Videostream wird in Unterkapitel 3.8 erläutert. Eine Darstellung der Ergebnisse aller Teilbereiche erfolgt in Kapitel 3.9.
Teilergebnisse dieses Kapitels wurden in zwei Publikationen veröffentlicht: einige Teile der Unterkapitel 3.4 bis 3.8 wurden in [87], Inhalte des Unterkapitels 3.8.2 in [85] veröffentlicht.

3.1 Grundlagen

3.1.1 Notation

Für die in dieser Arbeit beschriebenen Berechnungen werden folgende Annahmen getroffen:

- $\vec{x}$ ist der Ortsvektor des Punktes X.
- $\vec{AB}$ beschreibt den Vektor zwischen den Punkten A und B.
- Der Vektor $\vec{x}$ hat die Komponenten $(x_x, x_y, x_z)^T$.
- Der Betrag eines Vektors $\vec{x}$ ist definiert als $|\vec{x}| = \sqrt{x_x^2 + x_y^2 + x_z^2}$
- Ein normalisierter Vektor ist definiert als $\hat{x} = \frac{\vec{x}}{|\vec{x}|}$.
- $\vec{x} * \vec{y}$ beschreibt das Skalarprodukt zwischen den Vektoren $\vec{x}$ und $\vec{y}$.
- $\vec{x} \otimes \vec{y}$ beschreibt das Kreuzprodukt zwischen den Vektoren $\vec{x}$ und $\vec{y}$.
- $\vec{x}_{1,2}$ ist eine Kurzform für "$\vec{x}_1$ und $\vec{x}_2$".
- die Indizes 1 und 2 beziehen sich auf Kamera 1 und Kamera 2.

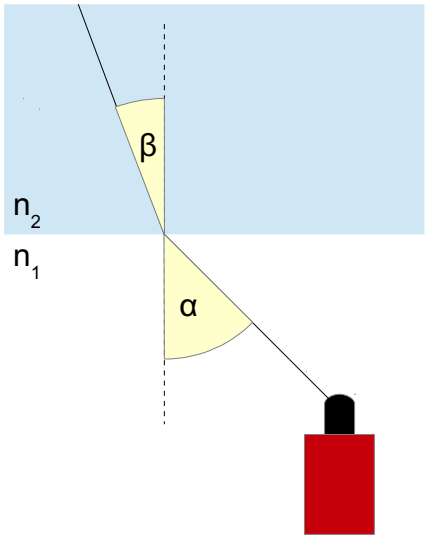


Abbildung 3.1: **Lichtbrechung.** Das Licht, welches in die Kamera (rot) einfällt, durchläuft zuvor zwei unterschiedliche Medien (Luft[n_1] und Wasser[n_2]) und wird an der Grenzfläche gebrochen. α gibt dabei den Winkel zwischen Kamerastrahl und β den Winkel zwischen gebrochenem Strahl und Orthogonale der Wasseroberfläche an.

3.1.2 Lichtbrechung

Das Video-Tracken von Fischen in einem Aquarium birgt eine besondere Schwierigkeit, die das Ergebnis verfälschen kann: die Lichtbrechung. Je nach Medium ändert sich die Phasengeschwindigkeit des Lichtes und es kommt am Übergang zwischen den Medien zur Lichtbrechung. Zur Berechnung der Brechung kann das Snelliussche Brechungsgesetz genutzt werden. Dieses beschreibt die Richtungsänderung der Ausbreitungsrichtung einer ebenen Welle beim Übergang von einem Medium ins andere. Zur Berechnung wird der Brechungsindex eines Mediums genutzt. Dieser dimensionslose Index gibt das Verhältnis der Lichtgeschwindigkeit im Vakuum c_0 zur Lichtgeschwindigkeit im jeweiligen Medium c_M an.

$$n = \frac{c_0}{c_M} \tag{3.1}$$

Der Brechungswinkel ist neben den Brechungsindizes auch vom Einfallswinkel des Lichtes abhängig. Je spitzer der Einfallswinkel ist, desto stärker wird das Licht gebrochen. Abbildung 3.1 zeigt den Übergang eines Lichtstrahles von einem Medium in ein anderes. Mit der Gleichung 3.2 kann die Brechung

beim Übergang vom ersten (Brechungsindex n_1) in das nächste Medium (Brechungsindex n_2) berechnet werden. α gibt dabei den Einfallswinkel des Lichtes an und β den Winkel des gebrochenen Strahles im Bezug zum Lot.

$$n_1 \; sin(\alpha) = n_2 \; sin(\beta) \tag{3.2}$$

3.1.3 Kamerageometrie und Abbildungsprozesse

Basis des Video-Trackings sind durch eine Kamera aufgenommene Bilder einer Szene. Bei der Aufnahme wird die dreidimensionale Szene auf die zweidimensionale Bildebene abgebildet. Um diesen Prozess zu beschreiben, wurden unterschiedliche *Kameramodelle* entwickelt. Mit Hilfe dieser Modelle können 3D-Punkte aus dem *Weltkoordinatensystem* der Szene über das *Kamerakoordinatensystem* in das zweidimensionale *Sensorkoordinatensystem* der Kamera-Bildebene und schließlich in Pixelkoordinaten (u, v) umgerechnet werden (siehe Abbildung 3.2). Zur Umrechnung eines Punktes aus dem Weltkoordinatensystem in Bildpixelkoordinaten müssen zwei Transformationen durchgeführt werden: mit Hilfe der *extrinsischen Kameraparameter* wird der Punkt aus dem Weltkoordinatensystem in das Kamerakoordinatensystem überführt. Die Transformation bestimmt die Position und Ausrichtung der Kamera in der Szene. Bei der zweiten Transformation werden mit Hilfe der *intrinsischen Kameraparameter* die 3D-Kamerakoordinaten in 2D-Koordinaten der Bildebene transformiert. Die Z-Achse des Kamerakoordinatensystems ist dabei orthogonal zur Bildebene ausgerichtet.

Die extrinsischen Kameraparameter bestehen zum einen aus der Kameraposition (t_x, t_y, t_z) und zum anderen aus den Rotationswinkeln (ω, ϕ, κ) der Kamera um die x-, y- und z-Achse des Weltkoordinatensystems. Diese Transformationen werden in der Kameramatrix K zusammengefasst:

$$K = \begin{bmatrix} r_{1x} & r_{2x} & r_{3x} & t_x \\ r_{1y} & r_{2y} & r_{3y} & t_y \\ r_{1z} & r_{2z} & r_{3z} & t_z \end{bmatrix} \quad \text{mit} \quad r_{\{1,2,3\},\{x,y,z\}} = r(\omega, \phi, \kappa) \tag{3.3}$$

Die Rotationsparameter $r_{\{1,2,3\},\{x,y,z\}}$ der Kameramatrix werden mit Hilfe von Winkelfunktionen aus den Drehwinkeln ω, ϕ und κ berechnet. Die genaue

Berechnung kann u. a. in der Arbeit von Klett et al. gefunden werden [61, S.189].

Die Umrechnung der Kamerakoordinaten (X_C, Y_C, Z_C) in Sensorkoordinaten (x, y) kann wiederum in zwei Unterschritte unterteilt werden. Durch die perspektivische Projektion werden die 3D-Koordinaten aus dem Kamerakoordinatensystem in 2D-Koordinaten der Bildebene überführt. Dazu werden die Gesetze der Zentralprojektion als übliche Näherung genutzt:

$$\frac{x}{X_C} = \frac{y}{Y_C} = \frac{f}{Z_C}. \tag{3.4}$$

Daraus ergibt sich

$$x = \frac{X_C f}{Z_C} \qquad \text{und} \qquad y = \frac{Y_C f}{Z_C}. \tag{3.5}$$

f bezeichnet dabei die Brennweite – die Entfernung vom Brennpunkt zum Kamerahauptpunkt. Die projizierten Koordinaten werden anschließend in diskrete Pixelkoordinaten überführt. Zum einen wird dabei der Urspung des Koordinatensystems von der Mitte des Sensors in die obere, linke Ecke verschoben. Diese Verschiebung wird durch die intrinsischen Kameraparameter u_c und v_c beschrieben. Zum anderen werden die Koordinaten in horizontaler und vertikaler Richtung auf die Größe der Bildebene in Pixel (Höhe (h) x Breite (b)) skaliert. Die Pixelkoordinaten (u, v) berechnen sich wie folgt:

$$u = u_c + \frac{x}{b} \qquad \text{und} \qquad v = v_c + \frac{y}{h}. \tag{3.6}$$

Aus den Gleichungen 3.5 und 3.6 ergeben sich

$$u = u_c + \frac{X_C f}{bZ} \qquad \text{und} \qquad v = v_c + \frac{Y_C f}{hZ}. \tag{3.7}$$

Dies lässt sich in die Projektionsmatrix P überführen, mit der Kamerakoordinaten in Pixelkoordinaten transformiert werden können.

$$P = \begin{bmatrix} \alpha_u & 0 & u_c \\ 0 & \alpha_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \quad \text{mit} \quad \alpha_u = \frac{f}{b} \quad \text{und} \quad \alpha_v = \frac{f}{h} \tag{3.8}$$

Ein Punkt $\tilde{M}$ aus dem Weltkoordinatensystem kann nun durch Kombination der Kameramatrix K und der Projektionsmatrix P in einen Punkt $\tilde{m}$ in

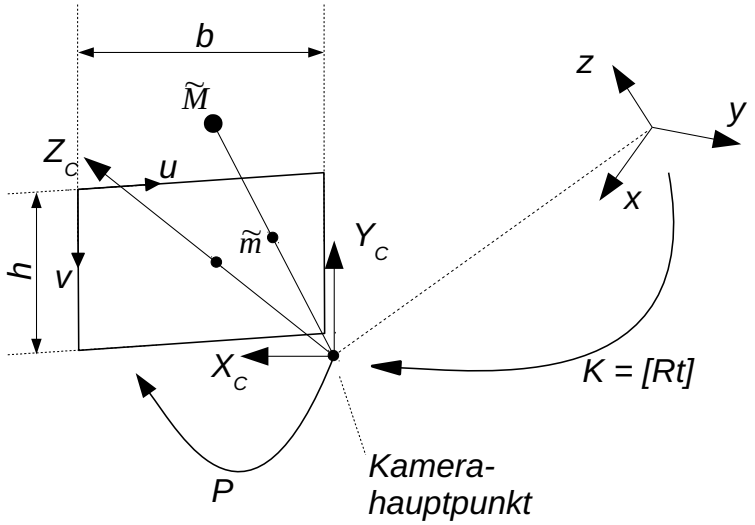


Abbildung 3.2: **Übersicht der Koordinatensysteme.** Mit der Kameramatrix K wird die Weltkoordinate $\tilde{M}$ in das Kamerakoordinatensystem überführt. Die Projektionsmatrix P transformiert die Kamerakoordianten in die Pixelkoordinate $\tilde{m}$.

.

Pixelkoordinaten überführt werden. Die aus der Multiplikation der Matrizen K und P entstandene Matrix C wird oft auch als Kamerakalibrierungsmatrix bezeichnet.

$$\tilde{m} = C * \tilde{M} \qquad \text{mit} \qquad C = P * K \tag{3.9}$$

Verzeichnung durch die Linse Die zuvor beschriebene Umrechnung von der Szene zum Bild basiert auf der Annahme, dass die Kamera einem idealen zentralperspektivischen Modell einer Lochkammer genügt. Die meisten Kameras sind jedoch mit Objektiven ausgestattet, die das Bild verzeichnen. Besonders bei Weitwinkelkameras wird dieser Effekt besonders deutlich: gerade Linien in der Szene werden im Bild gebogen dargestellt. Dieser Effekt wird **tonnenförmige** bzw. **kissenförmige Verzeichnung** genannt und ist in Abbildung 3.3 dargestellt (vgl. [133, S.357 f.]). Eine Möglichkeit diesen Effekt zu minimieren, ist der Einsatz von telezentrischen Objektiven. Diese teils hochpreisigen Objektive werden besonders in messtechnischen Anwendungen genutzt. Eine weitere Möglichkeit besteht darin, die Verzeichnung basierend auf

(a) tonnenförmige Verzeichnung

(b) verzeichnungsfreies Bild

(c) kissenförmige Verzeichnung

Abbildung 3.3: **Verzeichnung durch Linsen.** An Hand eines Aquarium-Bildes ist die Auswirkung der tonnen- und kissenförmigen Linsenverzeichnung exemplarisch dargestellt.

einem Modell abzuschätzen und zurückzurechnen. Zhang nutzt für die Korrektur von verzeichneten Koordinaten (x, y) hin zu korrigierten Koordinaten $(\tilde{x}, \tilde{y})$ ein Modell, bei dem die radiale Verzeichnung durch die Parameter k_1 und k_2 beschrieben wird und das Zentrum der Verzeichnung im Kamerahauptpunkt liegt [154].

$$\tilde{x} = x + x[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \tag{3.10}$$

$$\tilde{y} = y + y[k_1(x^2 + y^2) + k_2(x^2 + y^2)^2] \tag{3.11}$$

Die Parameter k_1 und k_2 bestimmt Zhang mit Hilfe eines Kalibrierprozesses, welcher im Detail unter [154] beschrieben ist.

3.1.4 Tracking

Aus dem Englischen übersetzt bedeutet "Tracking"*jemanden bzw. etwas verfolgen*. Beim Video-Tracking wird der Status (z.B. Position und Pose) eines oder mehrerer Objekte in einer Bildsequenz verfolgt (vgl. [70, S. xvii]). Als Grundlage des Video-Trackings muss somit initial bekannt sein, welche Objekte verfolgt werden sollen und wo sich diese befinden. Diese Informationen kann das Tracking-Verfahren auf unterschiedliche Arten erhalten: im Falle von **interaktiven Video-Tracking-Algorithmen** teilt der Benutzer dem System zu Beginn des Trackings mit, welche Objekte verfolgt werden sollen und wo sich diese befinden. Dies kann z. B. mit Hilfe einer →grafischen Benutzeroberfläche erfolgen, indem der Benutzer die Objekte mit einem Eingabegerät (z.B.

Maus) markiert. Im Gegensatz zum interaktiven Video-Tracking ist bei den **automatischen Video-Tracking** Verfahren kein manueller Eingriff erforderlich. Eine Möglichkeit der automatischen Objektverfolgung ist das **Marker-basierte Video-Tracking**. Bei dieser Methode werden an die zu verfolgenden Objekte Marker angebracht, die das Tracking-System automatisch aus dem Videobild extrahieren kann. Dazu können z. B. farbige Markierungen, Reflektoren oder auch Muster (z.B. → QR-Codes) genutzt werden. Um bei der automatischen Objektverfolgung nicht auf Marker angewiesen zu sein, bieten sich Verfahren aus der **Objektdetektion** an. Die Objektdetektion dient dem Auffinden von Objekten und deren Lokalisierung im Bild oder in Videosequenzen (vgl. [120, S. 1]). Einen Einblick in die Objektdetektion wird in Unterkapitel 3.1.4.1 gegeben. Um das Objekt im späteren Verlauf mit Hilfe des Tracking-Algorithmus zu verfolgen, muss eine digitale Objektapproximation gefunden werden, die es ermöglicht, die wichtigsten Objektmerkmale und den zu verfolgenden Status (z. B. Position, Pose, etc.) abzubilden. Die gängigsten Approximationen sind in Unterkapitel 3.1.4.2 dargestellt. Nach der Initialisierung des Tracking-Verfahrens bieten sich verschiedene Tracking-Algorithmen an, um das Objekt im weiteren Verlauf der Videosequenz zu verfolgen. Eine Übersicht grundlegender Tracking-Verfahren ist in Unterkapitel 3.1.4.3 zu finden. In Unterkapitel 3.1.4.4 werden das Mean-Shift- und das CamShift-Verfahren genauer beschrieben. Während des Trackings kann es zu Verdeckungen des zu trackenden Objektes kommen. Um auch in solchen Situationen die Pfadverfolgung nicht zu unterbrechen, werden Schätzverfahren angewendet, durch welche mögliche Bewegungen während der Verdeckungsphase berechnet werden. Eine kurze Einführung zu diesen Verfahren ist in Unterkapitel 3.1.4.5 gegeben. Das Themenfeld des Video-Trackings hat sich über viele Jahre in unterschiedliche Richtungen entwickelt. An dieser Stelle erfolgt eine grundlegende Einordnung, welche dem Verständnis der in der Arbeit verwendeten Verfahren dient und keinen Anspruch auf Vollständigkeit erhebt. Für weitere Informationen zum Thema sind die Arbeiten von Yilmaz et al.[153] und Maggio und Cavallaro[71] zu empfehlen.

3.1.4.1 Objektdetektion

Da die Objektdetektion ein großes und facettenreiches Forschungsgebiet darstellt (Detektoren wurden u. a. für Gesichter [144], menschliche Haut [143], Fußgänger [40], Autos [156], Vegetation [91] und Krebszellen [32] entwickelt), wird im Folgenden lediglich der Teilbereich der Objektdetektion betrachtet, der für das in der Arbeit beschriebene System relevant ist: die Detektion von bewegten Objekten in Videos. Doch auch bei einer Fokussierung auf diesen Teilbereich ist keine eindeutige und allgemeingültige Klassifizierung der Verfahren möglich. Oft werden mehrere Methoden kombiniert und auf den speziellen Anwendungsfall angepasst. Shaikh et al. [120, S. 5] haben in ihrer Arbeit eine Klassifizierung der Methoden in vier verschiedene Kategorien vorgenommen: Background Subtraction (Hintergrundsubtraktion), Statistical Approaches (Statistische Ansätze), Temporal Differencing (Zeitliche Differenzierung) und Optical Flow (Optischer Fluss). Im Folgenden werden die Kategorien kurz beschrieben. Da die statistischen Ansätze als Erweiterung der Hintergrundsubtraktion gesehen werden können, werden diese Kategorien zusammengefasst. **Background Subtraction und Statistical Approaches** sind schon seit langer Zeit Gegenstand der Forschung und Methoden, Varianten und Anwendungen dieser Verfahren wurden in knapp 10000 Publikationen beschrieben [13, S. V]. Grundgedanke dieser Verfahren ist die Aufteilung des Videobildes in Vordergrund und Hintergrund. Der Vordergrund bildet die bewegten und zumeist interessanten Bereiche (z. B. Objekte) des Bildes ab, wohingegen der Hintergrund die statische Umgebung markiert. Um den Vordergrund zu erhalten, wird der Hintergrund vom Eingangsbild der Kamera abgezogen. Dies setzt voraus, dass sich die Position und Ausrichtung der Kamera im Bezug zur Szene nicht verändert. Der Hintergrund kann auf unterschiedliche Weise modelliert werden. Gleichung 3.12 beschreibt, wie sich bei der klassischen Methode Vordergrundmaske F_t zum Zeitpunkt t an Pixelkoordinate (x, y) durch die Subtraktion des Hintergrunds B vom Eingangsbild I bestimmen lässt.

$$F_t(x,y) = \begin{cases} 1 & falls \quad |I_t(x,y) - B(x,y)| > Schwellwert \\ 0 & sonst \end{cases} \tag{3.12}$$

Da Bildsensoren nicht deterministisch sind und je nach Güte starkes $\rightarrow$ Bildrauschen auftreten kann, ist die einfache Subtraktion eines zuvor aufgenomme-

nen Hintergrundbildes fehleranfällig. Unter anderem aus diesem Grund wurden verschiedene Hintergrundmodelle entwickelt. Einfache Verfahren berechnen den Hintergrund auf Basis des Durchschnitt- oder des Median-Pixelwerts während der Initialisierung. Bei der späteren Subtraktion werden die so bestimmten Pixelwerte des Hintergrundes durch einen Pufferwert nach oben und unten erweitert. Somit kann ausgeschlossen werden, dass der Hintergrundbereich bei leichten Schwankungen der Pixelwerte (z. B. durch Sensorrauschen oder Helligkeitsänderungen) als Vordergrund detektiert werden. Diese Verfahren sind schlank und benötigen wenig Rechenleistung, sind jedoch für Szenen mit Belichtungsänderungen (wie z.B. im Außenbereich) ungeeignet. Für diesen Anwendungsfall erweisen sich **dynamische Hintergrundmodelle** als hilfreich, die über die Zeit aktualisiert werden. Eine einfache Möglichkeit ist die Modellierung des Hintergrundes mit Hilfe von Häufigkeitsverteilungen über die Zeit. Besonders die Gauß-Verteilung bietet sich für diesen Fall an und wurde in mehreren Verfahren benutzt (z. B. [58]).

Am Beispiel des Außenbereiches lässt sich eine weitere Herausforderung bei der Hintergrundsubtraktion zeigen: Bildbereiche, die sich durch wiederkehrende Ereignisse in der Szene zeitweise ändern, wie es z. B. bei schwankenden Bäumen der Fall ist. Für diesen Fall eignen sich **Cluster-basierte Methoden**. Im Gegensatz zu den zuvor beschriebenen Methoden wird nicht nur ein Wertebereich als Hintergrund definiert, sondern mehrere Bereiche, die wiederkehrende Änderungen im Hintergrund abbilden. Im Falle des schwankenden Baumes würden zwei verschiedene Wertebereiche (Cluster) in das Hintergrundmodell überführt: der Baum und die Szene, die durch den Baum verdeckt wird. Somit bleibt der Baum trotz Bewegung bei der späteren Subtraktion im Hintergrund. Ein Cluster-basiertes Verfahren ist das **Codebook-Model**, welches von Kim et al. vorgestellt wurde [59]. Bei diesem Verfahren wird jedes Pixel des Hintergrundes mit beliebig vielen Codewörtern beschrieben. Ein Codewort umfasst eine genaue Beschreibung des Hintergrundpixels in Form eines Farbwertbereiches und einer Helligkeitsschranke. Sollten unterschiedliche Objekte der Szene während der Initialisierung von diesem Pixel erfasst werden, wird für jedes dieser Objekte (insoweit sich diese in Farbe und/oder Helligkeit unterscheiden) ein neues Codewort angelegt. Bei der späteren Subtraktion wird überprüft, ob die Pixel des Eingangsbildes einem der Codewörter

zugeordnet werden können. Sollte dies nicht der Fall sein, wird das Pixel als Vordergrund markiert. Das Codebook-Verfahren zeichnet sich besonders durch effektive Speichernutzung und hohe Geschwindigkeit im Vergleich zu anderen Verfahren aus (vgl. [13, S. 1-22]). Auch mit Methoden, die auf der Gauß-Verteilung basieren, lassen sich wiederkehrende Änderungen abbilden. Dazu werden sogenannte **Gaussian-Mixture-Models** definiert, die den wechselnden Hintergrund als Mischverteilung in Form von ein oder mehreren Gaußverteilungen im Wertebereich beschreiben (z. B. [129]).
Eine weitere Herausforderung sind Schatten im Bild. Diese können sowohl von Vordergrund-Objekten als auch von Hintergrund-Objekten geworfen werden. Die meisten Verfahren zur Schattendetektion nutzen das Zusammenspiel von → Chrominanz und → Luminanz aus (vgl. [116]). Beschattete Bereiche im Bild verlieren zwar an Luminanz (Helligkeit sinkt), behalten zumeist jedoch ihren Chrominanzwert (Farbe bleibt gleich). Im → RGB-Farbmodell werden Luminanz und Chrominanz nur sehr schlecht separiert. Aus diesem Grund wird oft auf Farbräume zurückgegriffen, die nach Farbwert, Sättigung und Helligkeit unterscheiden, wie z.B. das → HSV- oder → YCbCr-Farbmodell.
Das **Temporal Differencing** hat zum Ziel bewegte Bereiche in einer Videosequenz zu erkennen, indem zeitlich aufeinander folgende → Frames pixelweise verglichen werden. Bereiche, die sich über die Zeit verändert haben, werden hervorgehoben. Dieses Verfahren wird oft bei Videosequenzen angewendet, die von bewegten Kameras aufgenommen wurden. Um die Bewegungen der Kamera nicht mit den Objektbewegungen zu vermischen, wird meist in einer Vorverarbeitung die Kamerabewegung geschätzt.
Weitere Probleme, die beim Temporal Differencing auftreten, sind zum einen die Fehldetektion von Hintergrundbereichen, die im vorherigen Frame durch das Objekt verdeckt wurden (im Englischen als ghost region bezeichnet) und zum anderen die Nicht-Detektion von unbewegten Objekten. Um diese Probleme zu umgehen, wurde das Temporal Differencing auch mit Verfahren der Background Subtraction kombiniert (siehe z. B. [34]). Anders als bei den zuvor beschriebenen Verfahren hat die Methode des **Optischen Flusses** zum Ziel alle Bewegungsinformationen im Bild in Form eines in die Bildebene projizierten Vektorfeldes der Geschwindigkeiten zu bestimmen. Um die Geschwindigkeitsvektoren zu berechnen, stehen verschiedene Methoden zur Verfügung, die auf

Basis von Farb- oder Grauwertgradienten (z. B. [53, 69]), Pixelwertdifferenzen oder auch Merkmalsdeskriptoren (z. B. [68]) die Bewegung eines einzelnen Objektpunktes oder Bildbereiches messen. Optischer Fluss-Verfahren können auch bei bewegtem Hintergrund oder bewegter Kamera angewendet werden, sind jedoch sehr rechenaufwendig und nur mit spezieller Hardware in Echtzeit anwendbar (vgl. [120]).

3.1.4.2 Objektapproximation

Bei der Objektapproximation handelt es sich um eine Approximation des realen Objektes auf eine bezüglich des gewünschten Tracking-Ergebnis angepasste, digitale Darstellung. Dabei wird durch die Objektapproximation zum einen die **Objektform** und zum anderen das visuelle **Erscheinungsbild** (engl. appearance) abgebildet. Oft werden beide auch in Kombination angewendet. Die Objektapproximation stellt dabei die Grundlage des Trackers dar und sollte genau genug sein, um eine eindeutige Objektzuordnung im Szenario zu ermöglichen. Gleichzeitig muss die Approximation jedoch Veränderungen die durch Beleuchtungsvariation, Skalierung, Rotation oder Verdeckung entstehen, abdecken, insoweit diese im Szenario auftreten können.

Approximation der Objektform Yilmaz et al. [153] und Maggio und Cavallaro [71, S. 72ff.] nehmen eine ähnliche Aufteilung der Objektform-Approximation vor. Bei Yilmaz et al. werden diese als *shape models* und bei Maggio und Cavallaro als *shape approximations* bezeichnet. Im Folgenden wird eine Einteilung der Objektform-Approximation in Anlehnung an Maggio und Cavallaro und auf diese Arbeit zugeschnitten dargestellt. Die Objektform-Approximation kann in drei Hauptgruppen eingeteilt werden: **einfache**, **verformbare** und **gelenkige** Objektform-Approximation. Bei der einfachen Approximation werden Objekte durch einen einzelnen Punkt (z. B. Schwerpunkt des Objektes, Abb. 3.4 a) oder mehrere Punkte, die an bestimmte Merkmale des Objektes gebunden sind, dargestellt. Häufiger werden aus dieser Gruppe jedoch geometrische Formen wie z. B. Rechtecke (Abb. 3.4 b) oder Ellipsen genutzt, die das Objekt möglichst eng einschließen. Als dreidimensionale Erweiterung werden auch geometrische Körper (Abb. 3.4 e) genutzt, um die Objektform zu beschreiben. Bei der Punktdarstellung lässt sich nur die translatorische Bewegung des Objektes verfolgen. Die Darstellung mit mehreren Punkten oder durch geo-

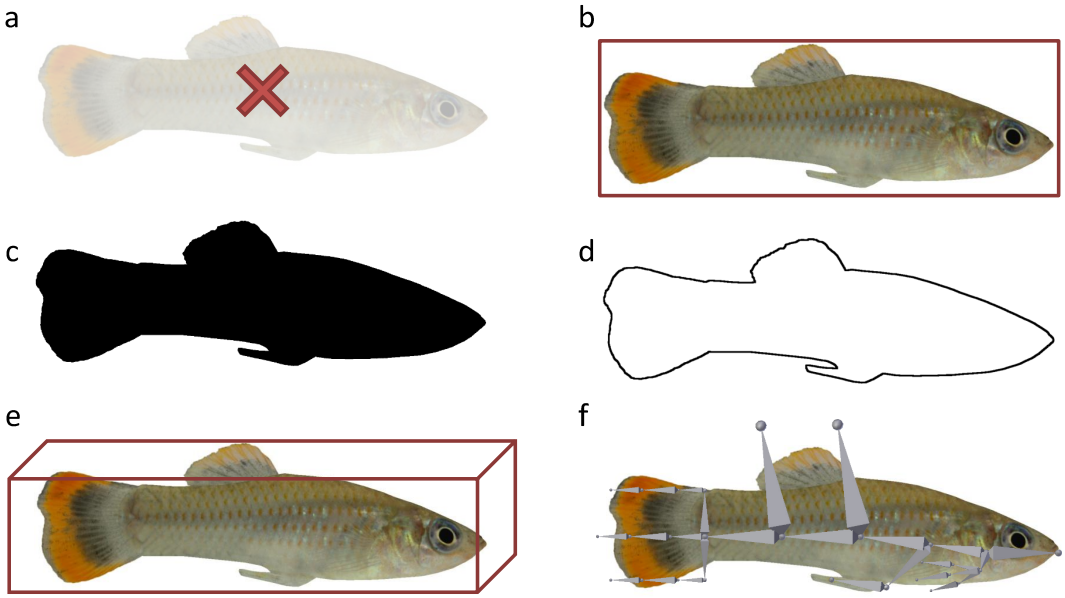


Abbildung 3.4: **Verschiedene Objektapproximationen anhand eines Fisches (Atlantik Kärpfling).** a) Punktapproximation b) Rechteckapproximation c) Silhouettenapproximation d) Konturapproximation e) 3D Würfelapproximation f) Gelenkapproximation

.

metrische Formen und Körper lassen zudem ein einfaches Posen-Tracking zu. Zu den verformbaren Approximationen zählen Objektkontur (Abb. 3.4 c) oder Objektsilhouette (Abb. 3.4 d). Gelenkige Objektform-Approximationen werden häufig bei der Darstellung von Lebewesen in Form von Knochenmodellen (Abb. 3.4 f) eingesetzt. Diese beinhalten neben der Objektposition auch Informationen über die Lage einzelner Körperteile (vgl. [153, S.3 ff]).

Approximation des visuellen Erscheinungsbildes Die Approximation des visuellen Objekterscheinungsbildes dient dazu, das Kamerabild des Objektes in einem Modell zu repräsentieren und das Objekt anhand dieses Modells im weiteren Verlauf des Videostreams zu verfolgen. Anders als bei der Approximation der Objektform werden bei der Approximation des visuellen Erscheinungsbildes die Pixelinformation (z. B. Farb- oder Grauwerte) des Objektes berücksichtigt. Maggio und Cavallaro unterteilen diese Art der Approximation in zwei Untergruppen: Schablonen (engl. template) und Histogramme (Farb-, Richtungs-, Strukturhistogramme)[71, S. 75ff.]. Yilmaz et al. neh-

men eine andere Unterteilung vor und nennen neben dem Schablonenmodell, Wahrscheinlichkeitsdichte-Modelle (engl. Density-Based Appearance Models), aktive, visuelle Modelle (engl. Active Appearance Models) und Mehransichtsmodelle (engl. Multiview Appearance Models) [153, S.4 ff]. Grundsätzlich ist eine Unterteilung der Verfahren nur schwer möglich, da es gängige Praxis ist, verschiedene Verfahren zu kombinieren. Für den weiteren Verlauf der Arbeit sind besonders Farbhistogramme als Approximation des visuellen Erscheinungsbildes interessant. Das Farbhistogramm wird auf Basis des markierten Objektes im Bild berechnet. Die Objektfläche wird meist mit Ellipsen oder Rechtecken beschrieben. Um Variationen in der Objektbeleuchtung besser abfangen zu können, nutzt man Farbmodelle, die auf dem → HSV-Farbraum basieren. Farbhistogramme können unter geringem Rechenaufwand erstellt werden und sind invariant gegen Skalierung und Rotation sowie Robust gegen Teilverdeckung (vgl. [71, S. 78f]).

3.1.4.3 Tracking-Algorithmen

Nach der Lokalisierung des zu verfolgenden Objektes ist es die Aufgabe des Tracking-Algorithmus' das Objekt über die anschließenden → Frames zu verfolgen und eine Trajektorie des Objektes zu erzeugen. Unter bestimmten Umständen kann auch die wiederholte Objektdetektion (das Objekt wird in jedem Frame erneut gesucht) zur Verfolgung genutzt werden (*tracking-by-detection*). Dabei sind jedoch folgende Einschränkungen zu beachten:

- Verfolgung von Individuen: Der Detektionsalgorithmus sucht in jedem Frame erneut nach Objekten. Ob das gefundene Objekt jedoch dasselbe Objekt wie im vorherigen Frame ist, kann der Algorithmus nicht feststellen. Tracking-Algorithmen versuchen basierend auf den zuvor gesammelten Informationen explizit einzelne Objekte zu verfolgen.

- Performance: Die Detektion ist in den meisten Fällen rechenintensiver als moderne Tracking-Algorithmen.

- Stabilität: Im Fall von Objektverdeckung bieten Tracking-Algorithmen oft die Möglichkeit basierend auf der vorherigen Objektposition und Geschwindigkeit die Position des Objektes zu schätzen (siehe hierzu auch

3.1.4.5). Die Objektdetektion würde in solch einem Fall kein Objekt finden.

Über die letzten Jahrzehnte wurden viele verschiedene Tracking-Algorithmen für unterschiedliche Anwendungen entwickelt. Der Tracking-Algorithmus ist im hohen Maße von der Art der Objektapproximation und von der Anzahl der Freiheitsgrade des Objektes abhängig. Yilmaz et al. [153] teilen das Objekt-Tracking in drei Bereiche ein: **Punkt-Tracking**, **Kernel-Tracking** und **Silhouette Tracking**.

Punkt-Tracking Beim Punkt-Tracking wird das Objekt lediglich durch einen Punkt repräsentiert und es kann ausschließlich die translatorische Bewegung verfolgt werden. Über das Objekt ist nur der Status (Position und Geschwindigkeit) aus den vorherigen Frames bekannt und es wird eine zusätzliche Methode benötigt, die in jeden Frame das potenzielle Objekt sucht. Yilmaz teil Punkt-Tracking-Methoden in deterministische und statistische Methoden auf. Bei den deterministischen Methoden werden Randbedingungen, wie z. B. örtliche Nähe, Geschwindigkeitsänderung und maximale Geschwindigkeit benutzt, um das Objekt im folgenden Frame wiederzufinden. Zu den statistischen Methoden werden statistische Schätzverfahren gezählt, welche genauer in Kapitel 3.1.4.5 behandelt werden. Punkt-Tracking-Verfahren werden auch für Aufgaben genutzt, bei denen mehrere Objekte gleichzeitig getrackt werden müssen. (vgl. [153, S. 16ff.])

Kernel-Tracking Anders als beim Punkt-Tracking sind beim Kernel-Tracking (engl. Kern) die zum Auffinden des Objektes benötigten Informationen durch den Kernel selbst repräsentiert. Als Objektapproximation wird zumeist das visuelle Erscheinungsbild in Form einer Objektschablone (Template) oder eines Objekthistogramms genutzt. Die Objektregion wird meist mittels Rechteck oder Ellipse umrandet. Neben der translatorischen Bewegung des Objektes werden je nach Algorithmus Skalierung, Rotation oder auch die affine Transformation getrackt. Die einfachste Form des Kernel-Trackings bildet der Schablonen-Vergleich. Dabei wird zu Beginn des Verfahren ein Bild des Objektes (die Schablone) manuell oder automatisch bestimmt. Im folgenden Frame wird der umliegende Bildbereich mittels → Brute-Force-Methode auf Ähnlich-

keiten mit der Objektschablone untersucht. Übersteigt die Ähnlichkeit einen Schwellwert, gilt der untersuchte Bildausschnitt als neue Position des Objektes. Der Schablonenvergleich ist auf Grund der eingesetzten →Brute-Force-Methode sehr rechenaufwendig. Aus diesem Grund wurden auch laufzeitoptimiertete Methoden entwickelt (siehe z. B. [119]). Als Beispiel für Histogrammbasierte Kernel-Tracking-Methoden sei das **Mean-Shift-Tracking** [28] und dessen Erweiterung, das **CamShift-Tracking** [14], genannt. Da das CamShift-Tracking auch in dieser Arbeit genutzt wird, gibt das Unterkapitel 3.1.4.4 eine kurze Einführung in die Funktionsweise. (vgl. [153, S. 17ff.])

Silhouetten-Tracking Beim Silhouetten-Tracking wird die Objektfläche im Bild durch die Objektkontur abgegrenzt. Dies ermöglicht im Vergleich zum Kernel-Tracking, bei dem die Abgrenzung durch geometrische Formen wie Rechtecke oder Ellipsen stattfindet, eine sehr genaue Beschreibung der Objektfläche. Die Objektkontur selbst wird meist durch eine vorangehende Hintergrundsubtraktion (siehe Unterkapitel 3.1.4.1) oder durch ein Kantendetektionsverfahren bereitgestellt. Yilmaz et al. [153] unterteilen die Silhouetten-Tracking-Verfahren in zwei Gruppen: zur einen Gruppe werden Verfahren gezählt, bei denen ein **Konturvergleich** durchgeführt wird. Die andere Gruppe umfasst Verfahren, die auf **Kontur-Tracking** basieren und die Entwicklung der Kontur von Frame zu Frame verfolgen.
Bei einfachen Konturvergleich-Verfahren wird, ausgehend von der Kontur aus dem vorherigen Frame, die Kontur im akutellen Frame gesucht. Dies geschieht, ähnlich wie beim Schablonenvergleich, iterativ. Die gefundene Kontur bildet die Grundlage für die Suche im nächsten Frame. Durch die stetige Reinitialisierung des Objekt-Kontur-Modells können auch Objekte verfolgt werden, die durch Rotation oder Verformung ihr Erscheinungsbild (Kontur) über die Zeit ändern.
Beim Kontur-Tracking wird im Gegensatz zum Konturvergleich ein initiales Konturmodell über die Zeit weiterentwickelt. Dies kann über dynamische Bewegungsmodelle der Konturpixel geschehen. Jedem Pixel (alternativ wurden auch zur Einsparung von Rechenzeit wenige Kontrollpunkte entlang der Kontur benutzt) wird ein eigenes Bewegungsmodell zugeordnet, mit welchem die voraussichtliche Position im nächsten Frame geschätzt werden kann (siehe Un-

terkapitel 3.1.4.5). Andere Ansätze führen eine Konturverfolgung durch die Minimierung von Kontur-Fehlerfunktionen durch. Diese Fehlerfunktionen beschreiben wie gut das Kontur-Modell auf die potenzielle Kontur im aktuellen Frame angewendet werden kann. (vgl. [153])

3.1.4.4 Mean-Shift- und CamShift-Tracking

Das Mean-Shift-Tracking und dessen Erweiterung das CamShift-Tracking (***Con**tinuously **A**daptive **M**ean Shift*) wurden in mehreren Tausend Publikationen zitiert und finden nach wie vor Anwendung. Beide sind Kernel-basierte Verfahren und nutzen die Objektfarbinformationen zur Verfolgung. Dies setzt voraus, dass sich das Objekt farblich von seiner Umgebung abhebt. Beim Mean-Shift-Tracking wird das Objekt durch ein Rechteck von konstanter, initial festgelegter Größe repräsentiert. Das CamShift-Verfahren passt zusätzlich die Größe und Ausrichtung des Rechtecks während des Trackings an. Neben den zusätzlichen Informationen, die während des Trackings durch CamShift gewonnen werden (Richtung der Objekt-Hauptachse und Objektgröße) ist das Verfahren besonders bei variierender Objektgröße in der Bildebene (z. B. durch Objektbewegung hin zur Kamera oder weg von der Kamera) stabiler als das Mean-Shift-Verfahren. Im Folgenden wird der Ablauf dieser Verfahren dargestellt. Da das CamShift- auf das Mean-Shift-Tracking aufsetzt, wird letzteres im ersten Teil beschrieben.

Mean-Shift-Tracking Grundlage beider Verfahren bildet ein Farbhistogramm des Objektes. Das Objektbild kann dabei in einem beliebigen Farbmodell vorliegen. Bradski [14] verwendet in seiner Arbeit das → HSV-Farbmodell, bei dem nach Farbwert, Farbsättigung und Hellwert unterschieden wird. Zur Histogrammberechnung nutzt er lediglich den Farbwert-Kanal des Bildes, der ausreichend Informationen für seinen Anwendungszweck, das Gesichtstracking, bietet. Nach Berechnung des Objekt-Farbhistogramms beginnt die Verfolgung des Objektes. Zur Initialisierung des Tracking-Verfahrens muss die Größe des Suchfensters, welches das Objekt umrandet, und dessen Position im Bild manuell oder automatisch bereitgestellt werden. Ausgehend von diesem initialen Suchfenster läuft der Algorithmus wie folgt ab:

1. Berechne die Stelle im Suchfenster mit der höchsten Dichte an hohen

Pixelwerten. (Die genaue Berechnung wird im nächsten Abschnitt beschrieben.)

2. Verschiebe das Zentrum des Suchfensters an die zuvor gefundene Stelle mit der höchsten Dichte.

3. Wiederhole Schritte 1 und 2 bis zur Konvergenz (die Verschiebung des Suchfensters ist geringer als eine definierte Schranke).

Zu der in Schritt 1 genannten Berechnung der Bildstelle mit der höchsten Dichte muss die Verteilung des Objekt-Histogramms im Suchfenster zurückprojiziert werden. Farben, die im initialen Histogramm häufig vertreten sind, werden in der Rückprojektion mit einem hohen Wert (im Bild hell) versehen; Farben, die nur selten oder nicht vorkommen, werden mit niedrigen Werten dargestellt (dunkel) (siehe Bild 3.5a). Zur Bestimmung der Position mit der höchsten Dichte hoher Pixelwerte im Suchfenster (heller Bereich im Bild), werden die zentralen Bildmomente M_{ij} herangezogen. $I(x, y)$ bildet dabei die Bildfunktion und liefert die Pixelwahrscheinlichkeitswerte an der Bildkoordinate (x, y).

$$M_{00} = \sum_x \sum_y I(x, y)$$

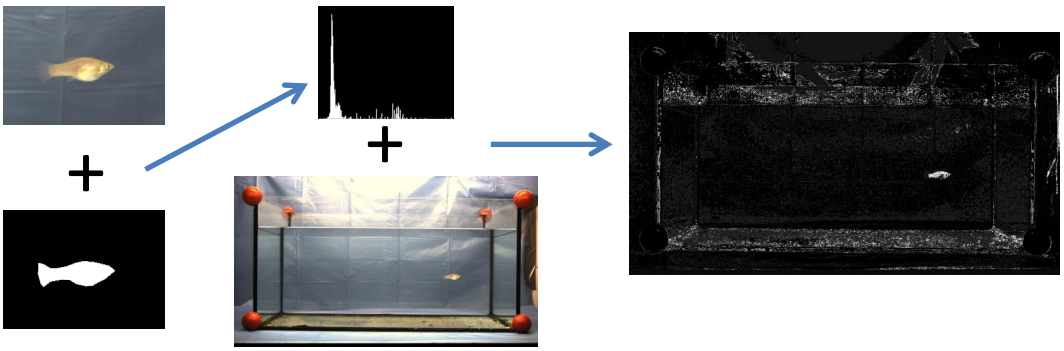
$$M_{10} = \sum_x \sum_y xI(x, y) \qquad M_{01} = \sum_x \sum_y yI(x, y) \tag{3.13}$$

$$S : \{x_s, y_s\} = \{\tfrac{M_{10}}{M_{00}}, \tfrac{M_{01}}{M_{00}}\}$$

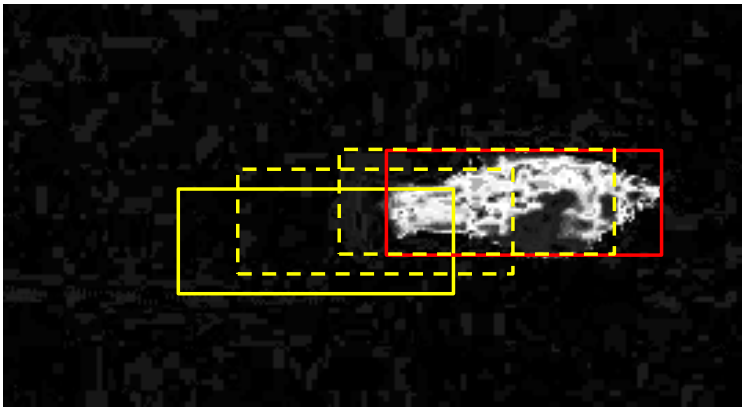
Das Suchfenster wird in Schritt 2 zur Position S geschoben. Dieses Verfahren wird bei jedem neuen Frame wiederholt. Das Suchfenster wird an der zuvor gefundenen Position initialisiert.

CamShift-Tracking Das Mean-Shift-Tracking bildet die Grundlage des CamShift-Verfahrens. Anders als beim Mean-Shift-Verfahren wird beim CamShift-Verfahren nach jedem Iterationsschritt (Verschiebung des Suchfensters) auch die Größe des Objektausschnittes angepasst. Der Ablauf des Algorithmus ändert sich somit wie folgt:

1. Berechne die Stelle im Suchfenster mit der höchsten Dichte.



(a) **Rückprojektion des Fisch-Histogramms.** Nach der initialen Erstellung eines Objekt-Histogramms werden die Farben aller folgenden Frames durch den Wert ihrer Häufigkeit im Histogramm ersetzt (Histogramm Rückprojektion).



(b) **Mean-Shift-Tracking.** Das initialisierte Suchfenster (gelbes Rechteck) wird iterativ an die neue Position (rotes Rechteck) geschoben. Die gestrichelten gelben Rechtecke symbolisieren die Zwischenschritte.

Abbildung 3.5: **Rückprojektion des Histogramms und Mean-Shift-Tracking.**

2. Verschiebe das Zentrum des Suchfensters an die zuvor gefunden Stelle mit der höchsten Dichte. Speicher M_{00} ab.

3. Wiederhole Schritte 1 und 2 bis zur Konvergenz (die Verschiebung des Suchfensters ist geringer als eine definierte Schranke).

4. Berechne die neue Suchfenstergröße mit Hilfe des gespeicherten Momentes M_{00}.

5. Wiederhole Schritte 2 bis 4 bis zur Konvergenz.

Die Berechnung der Suchfenstergröße (Breite b_F x Höhe h_F) basiert auf dem Bildmoment M_{00}. Bradski [14] nutzt für die Gesichtsverfolgung bei einem maximalen Pixelwert von 255 folgende Berechnung. Die Definition von h_F entspricht dem Verhältnisses von Höhe und Breite eines Gesichtes:

$$b_F = 2 * \sqrt{\frac{M_{00}}{255}}, \qquad h_F = 1.2 b_F \tag{3.14}$$

Nachdem die neue Objektposition im Bild gefunden wurde, werden beim CamShift-Verfahren zusätzlich die Ausrichtung Θ (siehe Gleichung 3.15) und die Größe des Objektes (Länge l_O und Breite b_O, siehe Gleichung 3.16) mit Hilfe weiterer Bildmomente berechnet.

$$M_{20} = \sum_x \sum_y x^2 I(x,y) \qquad M_{02} = \sum_x \sum_y y^2 I(x,y)$$

$$\Theta = \frac{arctan\left(\frac{2\left(\frac{M_{11}}{M_{00}} - x_s y_s\right)}{\left(\frac{M_{20}}{M_{00}} - x_s^2\right) - \left(\frac{M_{02}}{M_{00}} - y_s^2\right)}\right)}{2} \tag{3.15}$$

$$a = \frac{M_{20}}{M_{00}} - x_s^2 \qquad b = 2\left(\frac{M_{11}}{M_{00}} - x_s y_s\right) \qquad c = \frac{M_{02}}{M_{00}} - y_s^2$$

$$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}} \qquad w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}} \tag{3.16}$$

3.1.4.5 Bayes-Bewegungsschätzung

Das Tracken von Objekten unterliegt vielen potenziellen Stör- und Fehlerquellen. Diese können sowohl technischen (z. B. Bildsensorrauschen) als auch physikalischen Ursprungs (z.B. temporärer Objektverdeckung) sein. Jede Art von Störung kann je nach Tracking-Algorithmus zu Ungenauigkeiten, Abrissen der Objekttrajektorie oder Verlust des Objektes führen. Zudem kann es beim Tracken von mehreren Objekten zu Zuordnungsproblemen und somit zu Vertauschungen kommen.
Um diesem entgegenzuwirken, wird die rekursive Bayes-Schätzung genutzt, die im Folgenden in Anlehnung an die Arbeiten von Maggio und Cavallaro [71, S.95 ff.] und Yilmaz [153] dargestellt wird. Die rekursive Bayes-Schätzung schätzt den Systemzustand unter Zuhilfenahme eines Systemmodells und unter Berücksichtigung von Modell- und Messungenauigkeiten. Das Verfahren beruht auf einer Zustandsraummodellierung, welche die Dynamik des Systemzustands $X_k : k = 1, 2, ...$ und die Messung (im Falle des Video-Trackings z.B. Objektposition im Bild) $Z_k : k = 1, 2, ...$ unterscheidet. X_k ergibt sich dabei zum Zeitpunkt k aus der Zustandsgleichung f_k, welche die Übergänge zwischen zwei aufeinanderfolgenden Zuständen (X_{k-1} und X_k) beschreibt. Der Zusammenhang zwischen Systemzustand und Messung Z_k ergibt sich dabei aus der Mess- bzw. Beobachtungsgleichung g_k. Für beide Komponenten wird getrennt und voneinander unabhängig das Systemrauschen(m_k) bzw. das Messrauschen(n_k) definiert.

$$X_k = f_k(X_{k-1}, m_{k-1}) \tag{3.17}$$

$$Z_k = g_k(X_k, n_k) \tag{3.18}$$

Ziel des Verfahrens ist es den Systemzustand X_k unter Berücksichtigung aller bisher erfolgten Messungen $Z_{1...k}$ zu schätzen bzw. die Wahrscheinlichkeitsdichtefunktion $p(X_k|Z_{1...k})$ zu bestimmen. Die Schätzung erfolgt dabei rekursiv in zwei Schritten: dem Prädiktionsschritt und dem Filterschritt. Im Prädiktionsschritt wird das dynamische Bewegungsmodell (Gleichung 3.17) und die aus dem vorherigen Schritt $(k-1)$ berechnete Wahrscheinlichkeitsdichte genutzt, um die Apriori-Wahrscheinlichkeitsdichte

$$p(X_k|Z_{1...k-1}) = \int p(X_k|X_{k-1})p(X_{k-1}|Z_{1...k-1})dx_{k-1} \tag{3.19}$$

zu bestimmen. Diese beschreibt die Zustandswahrscheinlichkeit vor dem Eintreffen der nächsten Messung. Der Filterschritt hingegen berechnet die Aposteriori-Wahrscheinlichkeitsdichte

$$p(X_k|Z_{1...k}) = \frac{p(Z_k|X_k)p(X_k|Z_{1...k-1})}{p(Z_k|Z_{1...k-1})} \tag{3.20}$$

unter Zuhilfenahme der Wahrscheinlichkeitsdichte der aktuellen Messung $p(Z_k|X_k)$.

Kalman-Filter Da die Gleichungen 3.19 und 3.20 analytisch nicht lösbar sind (vgl. [71, S.96]), finden in der Praxis Vereinfachungen Anwendung. Eine weit verbreitete Lösung ist der Kalman-Filter [56], der unter anderem auch Anwendung beim Video-Tracking findet (vgl. [15]). Beim Kalman-Filter wird zur Vereinfachung angenommen, dass die Gleichungen 3.17 und 3.18 linear und $p(X_{k-1}|Z_{1...k-1})$ die Rauschparameter m_{k-1} und n_k normalverteilt sind. Daraus ergeben sich folgende Gleichungen:

$$Z_k = G_k X_k + n_k \tag{3.21}$$

$$X_{k|k-1} = F_k X_{k-1} \tag{3.22}$$

Die Matrix F_k beschreibt dabei die lineare Beziehung zwischen aufeinander folgende Zustände und G_k die lineare Beziehung zwischen Zustand und Messung. Auf Grund der Normalverteilung der Rauschparameter und der Linearität des zugrundeliegenden Modells, sind auch die gesuchten Zustände normalverteilt. Die Normalverteilung lässt sich durch ihren Mittelwert $\overline{X}_{k|k-1}$ und die Kovarianz $P_{k|k-1}$ beschreiben. Der Index $k|k-1$ beschreibt dabei die Abhängigkeit der Schätzungen zum Zeitpunkt k bzw. $k-1$.

$$\overline{X}_{k|k-1} = F_k \overline{X}_{k-1} \tag{3.23}$$

F_k^T beschreibt die transponierte Matrix F_k.

$$P_{k|k-1} = F_k P_{k-1} F_k^T + Q_k \tag{3.24}$$

R_k und Q_k sind die Kovarianzen der Rauschparameter n_k und m_{k-1}. Durch Gleichung 3.21 lässt sich die nächste Messung $\hat{Z}_k$ abschätzen.

$$\hat{Z}_k = G_k \overline{X}_{k|k-1}$$

Nach Erhalt der nächsten Messung kann das Residuum $\overline{R}_k$ und dessen Kovarianz S_k berechnet werden.

$$\overline{R}_k = Z_k - \hat{Z}_k$$

$$S_k = G_k P_{k|k-1} G_k^T + R_k$$

Daraus ergibt sich die Kalman-Matrix K_k.

$$K_k = P_{k|k-1} G_k^T S_k^{-1}$$

Die finalen, rekursiven Schätzungen $\overline{X}_k$ und P_k ergeben sich nach Korrektur durch den aktuellen Messwert.

$$\overline{X}_k = \overline{X}_{k|k-1} K_k \overline{R}_k \tag{3.25}$$

$$P_k = P_{k|k-1} - K_k S_k K_k^T \tag{3.26}$$

3.2 Anforderungen

Das Einsatzgebiet des hier vorgestellten Tracking-Verfahrens ist die Verhaltensforschung zur Partnerwahl von Fischen auf Grundlage von visuellen Reizen. Diese Art von Experimenten folgt festen Abläufen, die durch das Tracking-Verfahren nicht beeinflusst werden dürfen (z. B. [49, 117, 118, 149]). Demgegenüber steht der Anspruch mit dem Tracking-System möglichst viele Informationen über den Fisch, seine Bewegung und damit über sein Verhalten zu sammeln. Diese Informationen werden bei der Auswertung der biologischen Experimente benötigt. In der vorliegenden Arbeit werden sie darüber hinaus zur interaktiven Animation eines virtuellen Fisch-Stimulus genutzt. Die aus diesen Punkten hervorgehenden Anforderungen sind im Folgenden genannt.

3.2.1 Einhaltung der Standards bei Partnerwahl-Experimenten

Partnerwahl-Experimente bei Fischen folgen festen Regeln. Diese definieren zum einen die Umgebung der Fische während des Experimentes und zum anderen den Ablauf. Der Aufbau besteht aus einem großen Aquarium, in welchem der Testfisch schwimmt, und zwei ([49]) oder vier ([149]) kleinen Aquarien, welche an der linken und rechten Seite bündig an das große Aquarium anschließen. Diese Aquarien beinhalten Stimulus-Fische. Die Stimulus-Fische können vom Testfisch durch die Aquariumsscheibe beobachtet werden. Die Rückwand des großen Aquariums wird zumeist durch eine farbige Folie verdeckt. Der Boden des Aquariums wird durch eine Kies- oder Sandschicht abgedeckt ([49]).
Das Experiment beginnt mit einer Akklimatisierungsphase, in der sich die Fische an das Becken gewöhnen können. Der Testfisch wird anschließend in einen Plexiglaszylinder in der Mitte des großen Beckens gesperrt, aus dem er beide Seiten und somit beide Stimuli beobachten kann, ohne sich nähern zu können. Nach einer definierten Zeit wird er durch den Experimentator aus dem Zylinder befreit und kann sich den Stimulus-Fischen nähern. Die Zeit, in der sich der Testfisch vor einem der Stimuli aufhält, wird gemessen und gibt Auskunft über dessen Präferenz. Um Seitenpräferenzen auszuschließen, wird der Versuch beginnend mit der Platzierung des Testfisches im Zylinder mit vertauschten Stimuli durchgeführt (vgl. [49]).
Ziel des Tracking-Systems ist es die Bewegungen des Testfisches im großen Becken zu verfolgen. Durch den oben beschriebenen Aufbau sind lediglich **zwei Seiten (Oberseite und Frontseite) des Aquariums** nutzbar, um den Fisch mittels Kameras zu tracken. Diese Einschränkung gilt nicht nur für die Kameras sondern auch für das Beleuchtungssystem. Bei der Beleuchtung ist zudem darauf zu achten, dass die Beleuchtungsstärke sich nicht von der natürlichen unterscheidet und die Fische während des Versuches irritiert. Dieser Wert kann je nach Fischart variieren.
Im Hinblick auf das Tracking sind besonders die Platzierung im Zylinder und Freilassung des Testfisches zu beachten. Während dieser Zeit muss sich der Experimentator vor dem Aquarium aufhalten und manuell den Zylinder aus dem Becken holen. Dies führt zum einen zur **Verdeckung des Fisches** und

zum anderen zur Wellenbildung auf der Wasseroberfläche und folglich zu **ungeordneter Lichtbrechung**. Besonders in der Zeit nach Freilassung des Fisches sollte jedoch die Bewegung getrackt werden.
Die Bedienung des System muss **einfach und benutzerfreundlich** gestaltet sein, da der Experimentator während des Experimentes bereits durch die Versuchsausführung stark eingebunden ist. Die **Kalibrierung und Konfiguration des Systems sollte zudem weitestgehend automatisch** durchgeführt werden.

3.2.2 Zeitliche und räumliche Genauigkeit

Die benötigte Genauigkeit des Trackings hängt stark von der Anwendung ab. Bei der manuellen Auswertung von Partnerwahl-Experimenten werden Zonen vor den Stimuli-Becken definiert und die Aufenthaltszeit in der jeweiligen Zone gemessen. Diese manuelle Messung kann durchaus durch ein 2D-Trackingsystem substituiert werden, welches mit einer Genauigkeit im Zentimeterbereich arbeitet und die **2D Position** des Fisches (Projektion der Position auf der Bodenebene) verfolgt. Durch den Einsatz eines Trackingsystems lassen sich jedoch weitere Fischparameter extrahieren und somit die Auswertung bereichern. In den Versuchen, wie sie von Gierzwski et al. [49] durchgeführt wurden, hatten die Aquarien einen Wasserstand von ca. 25 bis 35 cm, was dem Fisch zusätzliche vertikale Bewegung ermöglicht. Um auch diese Bewegung zu erfassen, ist ein **3D-Trackingsystem** erforderlich.
Höhere Anforderungen stellt die interaktive Animation eines virtuellen Stimulus dar. Um diesen authentisch auf Bewegungen des Testfisches reagieren zu lassen, muss das System die 3D-Positionsdaten in $\rightarrow$ **Echtzeit** bereitstellen. Für die genannten Anwendungsfälle kann eine **Punktapproximation** (siehe Abbildung 3.4) des Fisches ausreichend sein. Für die Bewegungsrekonstruktion (siehe Kapitel 4 und 5), wie sie zur authentischen Animation eines Stimulus-Fisches erforderlich ist, reicht diese nicht mehr aus. Für diesen Fall bietet sich die **Konturapproximation** an, welche je nach Ansicht Rückschlüsse über die **Biegung und Ausrichtung des Fisches** zulässt. Um unabhängig von der Ausrichtung des Fisches die Körperbiegung erfassen zu können, sind mindestens zwei orthogonal zueinander angebrachte Kameras erforderlich.

3.2.3 Spezialfall Aquarium

Neben den sonst üblichen Herausforderungen, die an ein Trackingsystem gestellt werden wie z.B. Verdeckung, Schatten oder Bildrauschen (vgl. [120, S.8 ff.], stellt das Tracken von Objekten in Aquarien besondere Anforderungen an das System. Diese treten in Form von **Brechung** und **Spiegelung** auf. Da das Aquarium mit Wasser gefüllt ist, tritt an der Grenzfläche zwischen Wasser und Luft bzw. Wasser, Glas und Luft eine Lichtbrechung auf. Diese kann je nach Blickwinkel der Kamera auf das Aquarium Verschiebungen im Bild erzeugen und somit die Ergebnisse verfälschen. Ein kleiner Teil des Lichtes wird zudem an der Aquariumscheibe reflektiert. Dies führt zu Spiegelungen des Objektes.

3.3 Stand der Technik

Durch die stetige Weiterentwicklung von Bildverarbeitungssystemen im Allgemeinen und Tracking-Systemen im Speziellen hat auch das Video-Tracking von Fischen Anwendung in viele Bereichen der Forschung gefunden. Neben der in dieser Arbeit thematisierten Verhaltensforschung wurden Fisch-Tracking-Systeme unter anderem in der Ökotoxikologie [39, 55, 63], in der Kognitionswissenschaft [12], in der Pharmakologie [23, 100] oder in der Tierwohlforschung [101] eingesetzt. Dabei substituieren Tracking- und Bildverarbeitungssysteme manuell durchgeführte, arbeitsintensive Aufgaben wie Fischzählung [79], Messung von Aufenthaltszeiten [92], die Beobachtung von Fischschwärmen [115, 134] oder Lebewesen in Korallenriffen [45]. Sie unterstützen bei der Fisch-Verhaltensanalyse [4, 97] oder eröffneten neue Möglichkeiten bei der Rekonstruktion der Schwimmkinematik [21].
Die Mehrzahl der eingesetzten Systeme zeichneten die Fischbewegung zweidimensional (2D) auf. Dies liegt zum einen an der Tatsache, dass 2D-Systeme im Vergleich zu 3D-Systemen mit erheblich geringerem Aufwand installiert und betrieben werden können. Zum anderen sind kommerzielle Lösungen zum 2D-Tracking in der Forschungsgemeinde weit verbreitet und auch durch fachfremde Anwender einfach zu benutzen. Die laut dem Hersteller [93] meist zitierte Software in diesem Bereich ist *EthoVision®* des Herstellers *Noldus Information Technology* (Wageningen, Niederlande).
Um die Bewegung der Testfische in der dritten Dimension zu begrenzen, wer-

den die Testbecken teils nur zu einem kleinen Teil mit Wasser gefüllt. Saberioon und Cisar [114] geben in ihrer Arbeit an, dass die meisten der Algorithmen eine Wasserhöhe von weniger als 5 cm benötigen. Somit können die Fische nur geringfügig ihre Positionshöhe ändern und die Reduktion von drei auf zwei Dimensionen verursacht keine großen Ungenauigkeiten. Neben einer höheren Positionsgenauigkeit bei gleichzeitig größerer Fisch-Bewegungsfreiheit bieten 3D-Tracking-Systeme weitere Vorteile: Objektverdeckungen, die mit der Anzahl der Fische bzw. Objekte (z. B. Pflanzen oder Steine) im Aquarium zunehmen, können besser kompensiert werden.
Im Bereich des 3D-Fisch-Trackings sind in der Vergangenheit verschiedene Systeme eingesetzt worden. Laurel et al. [67] entwickelten ein System, dass den Schatten der Fische nutzte. Dazu wurden zwei Lichtquellen über einem Aquarium installiert und eine Kamera nahm sowohl den Fisch als auch dessen Schattenwurf auf. Aus diesen Informationen wurden mittels einer trigonometrischen Funktion die 3D-Positionen bestimmt. Auf Grund der Schattenüberschneidung ist diese Methode nicht für mehrere Fische geeignet.
Zhu und Weng [157] installierten im 45° Winkel Spiegel über und links neben dem Aquarium. Die vor dem Aquarium platzierte Kamera nahm neben der Frontansicht auch die Seitenansicht (links) und die Aufsicht mit Hilfe der Spiegel auf. Dies bringt den Vorteil, dass durch eine Kamera mehrere Ansichten aufgenommen werden können. Als Nachteil dieser Lösung ist die geringere Auflösung und der höhere Installationsaufwand zu nennen. Die meisten 3D-Tracking-Systeme nutzen mehrere Kameras. Viscido et al. [145] nutzten zwei Camcorder, welche über und vor dem Aquarium angebracht waren, um Gruppen von vier bzw. acht Malabarbärblingen (*Danio aequipinnatus*) zu verfolgen. Zur Rekonstruktion der 3D-Trajektorien wurden die 2D-Trajektorien im ersten Schritt in jedem Video getrennt voneinander mit Hilfe eines einfachen Bildverarbeitungsprogramms extrahiert, überprüft und manuell überarbeitet. Im zweiten Schritt nutzten sie ein weiteres Programm, welches die 2D-Trajektorien zu einer 3D-Trajektorie vereinte.
Cachat et al. [24] nutzten in ihrer Arbeit zwei Kameras, die in selber Konfiguration wie bei Viscido et al. angebracht waren. Der 2D-Pfad des aufgenommenen Zebrafisches (*Danio rerio*) wurde mit Hilfe der 2D-Tracking-Software (*EthoVisionXT®*) für jede Kameraansicht separat ausgewertet und manuell

zur 3D-Trajektorie vereint.
Mittlerweile wird auch kommerzielle 3D-Tracking-Software angeboten. *Noldus Information Technologies* bietet z. B. die Erweiterung *Track3D* für die Tracking-Software *EthoVisionXT* an [94]. Stewart et al. [130] setzten diese Methode ein, um das Bewegungssystem von Zebrafischen (*Danio rerio*) unter Einfluss von Psychopharmaka zu analysieren. Ähnlich wie bei Cachat et al. [24] wird bei dieser Methode die 2D-Trakjektorie für jede Ansicht separat mit einer EthoVisionXT Instanz ermittelt und anschließend mit Hilfe des Track3D Tools zu einer 3D-Trajektorie vereint. Butail und Paley setzten in ihrer Arbeit zur Rekonstruktion der Kinematik von Schwarmfischen drei orthogonal zueinander ausgerichtete Kameras ein, welche vor, neben und über dem Becken angebracht waren [21]. Sie rekonstruierten Fischbewegungen von bis zu acht Malabarbärblingen (*Danio aequipinnatus*) auf der Basis von Fisch-Silhouetten. Auch Stereo-Kamera-Systeme mit zwei nebeneinander angebrachten und gleich ausgerichteten Kameras wurden zum Fisch-Tracking eingesetzt. Diese bieten den Vorteil bei kleiner Systemabmessung größere Bereiche (im Vergleich zu der orthogonalen Ausrichtung) überwachen zu können. Dieser Vorteil geht mit einer geringeren Tiefenauflösung und eingeschränkter Objektansicht (meist nur Seitenansicht des Fisches) einher. Aus diesem Grund wurden Stereo-Kamera-Systeme zumeist in offenen Gewässern (z. B. [31]) eingesetzt, fanden jedoch auch beim Tracking in Aquarien Anwendung (z. B. [52]). Um einen größeren Bereich zu überwachen, setzten Yamashita et al. [152] omnidirektionale Kameras ein, die übereinander angebracht waren und eine 360°-Ansicht ermöglichten.
Neben der 3D-Positionsberechnung aus mehreren Ansichten wurden auch Sensoren eingesetzt, die auf anderen Verfahren basieren. Saberioon und Cisar [114] setzten einen → RGB-D-Sensor ein, der mit strukturiertem Licht arbeitet (Microsoft Kinect). Dieser zeichnet neben den Farbinformationen ein → Tiefenbild der Szene auf. Saberioon und Cisar nutzten den Sensor, um mehrere Nil-Tilapien *(Oreochromis niloticus)* im Dreidimensionalen zu tracken. Im ersten Schritt wurde aus dem 2D-Farbbild basierend auf einem Schwellwert der homogene Hintergrund segmentiert und die Fische im Vordergrund extrahiert. Da jedem Farbpixel des RGB-D-Sensors ein Tiefenwert zugeordnet ist, kann nach Extraktion des Fisches im Farbbild die 3D-Position berechnet

werden. Zur Validierung der Tracking-Ergebnisse verglichen sie die Ergebnisse des Systems mit dem eines Stereo-Kamera-Systems, welches dieselbe Ausrichtung wie der 3D-Sensor hatte. Es konnte eine ähnliche Genauigkeit wie beim Stereo-Kamera-System erzielt werden, bei reduziertem Kalibrationsaufwand und Echtzeitfähigkeit.
In der hier präsentierten Arbeit werden zwei orthogonal zueinander ausgerichtete Kameras verwendet. Diese sind synchronisiert und nehmen ein vergleichsweise hochaufgelöstes Bild (1920 x 1080 Pixel) auf, welches eine präzise Extraktion der Fisch-Silhouette ermöglicht. Die extrahierten Silhouetten werden in Echtzeit in eine 3D-Abstraktion des Fisches (Kastenfisch) überführt und können in einer Simulationsumgebung während der Aufnahme dargestellt werden.
Das System zeichnet sich im Weiteren durch einen hohen Automatisierungsgrad aus: neben der Kalibrierung der Kameras funktioniert auch die Initialisierung des Tracking-Systems automatisch.
Zudem umfasst es Methoden, die auch unter widrigen Umständen, wie welliger Wasseroberfläche, Verdeckung oder Spiegelung Fische verfolgen können.

3.3.1 Echtzeitfähigkeit

Die Echtzeitfähigkeit des Fisch-Tracking-Systems ist in den meisten Fällen nicht erforderlich, da die Auswertung meist nachgelagert durchgeführt wird. Aus diesem Grund werden die Kameraaufnahmen meistens abgespeichert und im Nachgang ausgewertet. Im hier beschriebenen Szenario zur interaktiven Steuerung muss hingegen die Echtzeitfähigkeit gewährleistet werden, um auf die Bewegungen des Fisches reagieren zu können. Nur wenige Arbeiten beschäftigen sich mit Echtzeit-Fisch-Tracking-Systemen. Butkowski et al. [22] setzten ein echtzeitfähiges 2D-Tracking-System zum Starten einer virtuellen Fisch-Animation ein. In Zusammenarbeit mit der Firma *Biobserve GmbH* wurde ein Schwerträger (*Xiphophorus birchmanni*) mit der kommerziellen Tracking-Software *BIOBSERVE Viewer* [11] in Echtzeit getrackt [22]. Landgraf et al. [65] setzten ebenfalls ein 2D-Tracking-System ein, welches die Positionsdaten eines Guppy-Fischschwarms (*Poecilia reticulata*) in Echtzeit berechnete. Ein Roboter-Fisch folgte dem Fischschwarm bzw. leitete diesen basierend auf die Positionsinformationen.

Auch 3D-Tracking-Systeme mit Echtzeitfähigkeit wurden eingesetzt: Ho und Shish [52] nutzten ein Stereo-Kamera-System um Goldfische (*Carassius gibelio forma auratus*) in einem kleinen Aquarium zu tracken. Die Echtzeit-Positionsinformationen wurden genutzt, um die Fische mit einem 5-Achsen-Roboterarm mit Kescher einzufangen. Die beiden Kameras hatten eine geringe Auflösung von 320 x 240 Pixel und waren oberhalb des kleinen Aquariums (30 x 20 x 18 cm^3) angebracht.
Butail, Chicoli and Paley [19] entwickelten ein Virtual-Reality-System für einen Malabarbärbling (*Danio aequipinnatus*). Dazu wurde ähnlich wie bei Zhu und Weng [157] eine Kamera und ein Spiegel eingesetzt, um zwei Ansichten des Fisches zu erhalten. Aus diesen zwei Ansichten wurde eine 3D-Position des Fisches extrahiert und in Echtzeit berechnet. Die so gewonnenen Informationen wurden genutzt, um in einer 3D-Animation ein elliptische Scheibe auf den Fisch zuzubewegen und dabei die Ansicht auf die Fischposition zu optimieren. Das im Rahmen dieser Arbeit entwickelte System umfasst Methoden zum echtzeitfähigen 3D-Positionstracking für die interaktive Fischstimulus-Animation. Darüber hinaus ist es in der Lage nicht nur die 3D-Position in Echtzeit bereitzustellen, sondern kann zudem auch die Konturen aus zwei Ansichten zu einem 3D-Objekt in Echtzeit vereinen.

3.3.2 Kompensation der Lichtbrechung

Beim 3D-Tracken von Fischen in Aquarien kann die Positionsberechnung bei nicht Berücksichtigung der Lichtbrechung mit großen Ungenauigkeiten behaftet sein. Zudem erschwert es die Zuordnung von Fischen in den verschiedenen Ansichten. Eine Möglichkeit den durch die Lichtbrechung verursachten Fehler zu minimieren ist die Kamerakalibrierung mit Hilfe eines unter Wasser aufgenommenen Kalibrierobjektes. Bei dieser Methode werden gängige Verfahren der Kamerakalibrierung benutzt (z.B. [154]), indem das Kalibrierobjekt in das Aquarium eingetaucht wird und die Kameras dieses von außen aufnehmen. Da diese Kalibriermethode die Lichtbrechung nicht berücksichtigt, sondern lediglich die Parameter eines Kameramodells und die Verzeichnungsparameter einer Linse schätzt, wird die Lichtbrechung während der Kalibrierung durch Verschiebung der Kameraposition, Verschiebung des Kamerabrennpunktes und durch Anpassung der Verzeichnungsparameter im Kameramodell kompensiert.

Dies führt zwar zu einer deutlichen Verbesserung des Ergebnisse im Vergleich zu Methoden, die eine Kamerakalibrierung ohne Wasser durchführen (vgl. [99]), eine eindeutige extrinsische Kalibrierung der Kamera wie in Abbildung 3.6 zu sehen erfolgt jedoch nicht.

Diese Methode fand unter anderem Anwendung in der Arbeit von Butail, Chicoli and Paley [19] und wird auch in der kommerziellen Tracking-Software Track3D verwendet [94]. Da diese Methode besonders in den Randbereichen, in denen die Brechung besonders stark ist, zu Fehlern führt, haben Kawahara, Nobuhara und Matsuyama in ihrer Arbeit ein Kameramodel (pixel-wise varifocal camera model) vorgestellt, welches für jedes Pixel eine auf die Lichtbrechung angepasste Brennweite beinhaltet, um so die Fehler, die durch die Brechung entstehen zu kompensieren [57]. Im Gegensatz dazu wird in dieser und anderen Arbeiten eine andere Methode genutzt, bei der die Lichtbrechung mit Hilfe der Brechungsgesetze modelliert wird. Dazu wird mit Hilfe der Lichtstrahlenverfolgung (engl. ray tracing) der Lichtstrahl vom Objekt bis zum $\rightarrow$ Sensel der Kamera modelliert. Yamashita et al. [152] nutzten dieses Verfahren in einem Stereoaufbau mit omnidirektionalen Kameras, die in einem wasserdichten, durchsichtigen Zylinder übereinander angeordnet waren und für den zukünftigen Einsatz an Unterwasserrobotern konzipiert wurden. Henrion et al. [50] setzten die Ray-Tracing-Methode für die Bewegungsanalyse eines Seepferdchens (*Hippocampus abdominalis*) ein. Zur Kalibrierung der Kamera wurde ein spezieller Würfel entwickelt, der in das Aquarium eingetaucht wurde. Anders als bei dem zuvor beschriebenen Verfahren werden bei der Schätzung der intrinsischen und extrinsischen Kameraparameter zusätzlich die Position und die Ausrichtung der Aquariumscheiben geschätzt. Die oberen und unteren Grenzen des Schätzraumes werden dabei manuell bestimmt. Pedersen et al. [99] verglichen in ihrer Arbeit die zwei oben genannten Methoden miteinander. Die Ray-Tracing Methode konnte neben einer höheren Genauigkeit besonders durch Flexibilität und Benutzerfreundlichkeit überzeugen.
Soweit dem Autor bekannt, wurde im Rahmen dieser Arbeit erstmals (zum Zeitpunkt der Veröffentlichung von [87]) ein Fisch-Tracking-System entwickelt, welches die Lichtbrechung durch Berechnung des Strahlenganges kompensiert und somit eine hohe Präzision erreicht. Zudem wurde ein Kalibrierungsverfahren entwickelt, welches automatisch die zur Berechnung der Brechung benö-

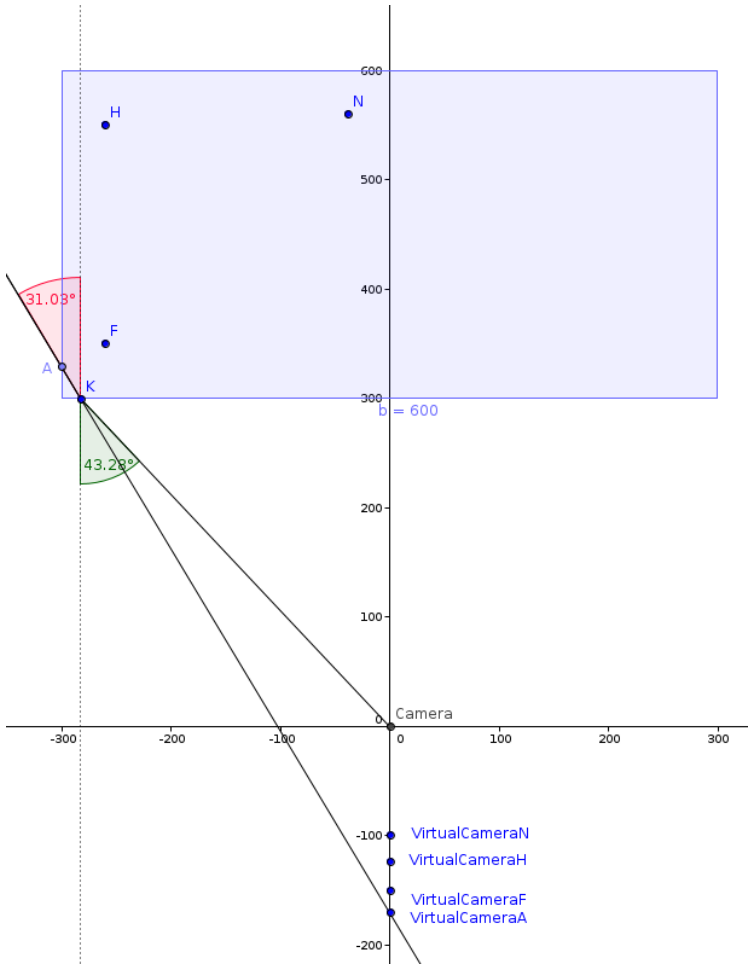


Abbildung 3.6: **Extrinsische Kamerakalibrierung mit Kalibrierungsobjekt im gefüllten Aquarium.** Da die Kalibriermethode die Lichtbrechung lediglich durch Verschiebung der Kameraposition kompensiert, gibt es keine eindeutige Lösung. Die Objektpunkte A, F, H und N im mit Wasser gefüllten Aquarium führen zu unterschiedlichen Positionen der virtuellen Kamera. Die Grafik wurde vorab in [87] veröffentlicht.

tigten Parameter (Kamerakalibrierungsmatrix und Brechungsebenen) anhand von Markern ermitteln kann.

3.4 Kamerakalibrierung

Die Kalibrierung der Kameras ist beim 3D-Tracken von Fischen essentiell. Besonders für die 3D-Bewegungsrekonstruktion (Kapitel 5) ist eine genaue Kalibrierung wichtig. Gleichzeitig muss das System eine einfache und benutzerfreundliche Bedienung bieten (vgl. Unterkapitel 3.2.1). Aus diesem Grund wird in dieser Arbeit eine zweistufige Kalibrierung entwickelt, die zudem die Grundlage bietet, die Lichtbrechung durch einen Ray-Tracing-Ansatz zu berücksichtigen. Im ersten Teil der Kalibrierung werden die intrinsischen Kameraparameter geschätzt. Da sich diese während des Systemeinsatzes nicht ändern, ist dieser Prozess vorgelagert und wird einmalig unter Laborbedingungen durchgeführt. Die zweite Stufe der Kalibrierung umfasst die weitgehend automatische, extrinsische Kalibrierung, die neben der Position und Ausrichtung der Kamera zusätzlich die Position der Grenzflächen zwischen Glas, Wasser und Luft bestimmt. Dieser Schritt muss nach jeder Änderung der Position oder Ausrichtung der Kamera oder des Aquariums durchgeführt werden und ist somit Aufgabe des Experimentators. Sollte z. B. während einer Versuchsreihe die Kamera versehentlich verschoben werden, kann die Kalibrierung mit Hilfe dieses Verfahrens einfach und schnell durchgeführt werden. Diese Methode ist in Unterkapitel 3.4.1 beschrieben.
Zur intrinsischen Kalibrierung und Schätzung der radialen Verzeichnungsparameter wird das von Zhang [154] entwickelte Verfahren benutzt. Mit Hilfe dieses Verfahrens können die Parameter auf Basis von 3D-Weltkoordinaten und den dazugehörigen 2D-Projektionspunkten auf der Bildebene geschätzt werden. Die Programmbibliothek für Bildverarbeitung *OpenCV* [96] bietet zur Generierung der 2D-3D-Punktpaare eine Methode zur automatischen Extraktion aus einem Schachbrettmuster mit bekannten Abmaßen an. Dieses wird aus mehreren Perspektiven aufgenommen, um möglichst viele Punktpaare aus verschiedenen Ansichten zu erhalten. Zur Erhöhung der Genauigkeit wird im letzten Schritt eine globale Parameteroptimierung mit Hilfe des Levenberg-Marquardt-Optimierungsverfahrens (siehe [80]) durchgeführt. Dazu werden die

3D-Weltkoordinaten auf Basis der berechneten Parameter zurückprojiziert und der Reprojektionsfehler in Form der quadrierten Entfernung minimiert. Als Ergebnis liefert die Methode die Projektionsmatrix P (vgl. 3.1.3).

3.4.1 Extrinsische Kalibrierung und Bestimmung der Grenzflächen zur Lichtbrechungsberechnung

Die extrinsische Kalibrierung der Kamera bei gegebener intrinsischer Kalibrierungsmatrix ist ein bekanntes Problem und kann mit Hilfe von Welt-Bild-Koordinatenpaaren gelöst werden. In der Literatur ist das Problem auch als PnP-Problem (engl. Perspective-n-Point) bekannt. Zur eindeutigen Lösung sind mindestens vier 2D-3D-Punktpaare erforderlich. Diese Punktpaare werden in der Bildebene durch Pixelkoordinaten und im Weltkoordinatensystem durch 3D-Koordinaten repräsentiert. Oft wird zur Bestimmung der Koordinatenpaare ein definiertes Kalibrierobjekt benutzt, dessen Abmessung bekannt ist und welches in der Bildebene einfach aufgefunden werden kann. Im hier beschriebenen Projekt wird bewusst auf ein mobiles Kalibrierobjekt verzichtet. Stattdessen wird das Aquarium selbst zur Kalibrierung genutzt. Dazu werden die Ecken mit einfach aufzufindenden Markern ausgestattet und der Ursprung des Weltkoordinatensystems in die linke, obere Ecke des Aquariums gelegt (siehe Abbildung 3.7). Dies hat zum einen den Vorteil, dass kein zusätzliches Kalibrierobjekt während des Versuches manuell in eine für alle Kameras sichtbare Position gebracht und ausgerichtet werden muss. Zum anderen ist es durch die fest mit dem Aquarium verbundenen Marker möglich, gleichzeitig die Grenzflächen der Lichtbrechung zu bestimmen. Die xy-Ebene bildet die Frontscheibe des Aquariums ab. Die xz-Ebene bildet die Wasseroberfläche bzw. eine Parallelebene zu dieser ab. Somit bietet die hier vorgestellte Kalibrierung über die Ecken des Aquariums nicht nur die Grundlage für die extrinsische Kalibrierung, sondern dienen zugleich der Lichtbrechungsberechnung. Zur eigentlichen Berechnung der extrinsischen Kalibrierung bzw. zur Lösung des PnP-Problems wurde die von Gao et al. [47] vorgestellte analytische Methode verwendet. Diese nutzt einen algebraischen Ansatz, um das PnP-Problem zu lösen. Grundsätzlich können an dieser Stelle auch anderen PnP-Algorithmen genutzt werden, die auf Basis von vier Punktkorrespondenzen die Position und Orientierung

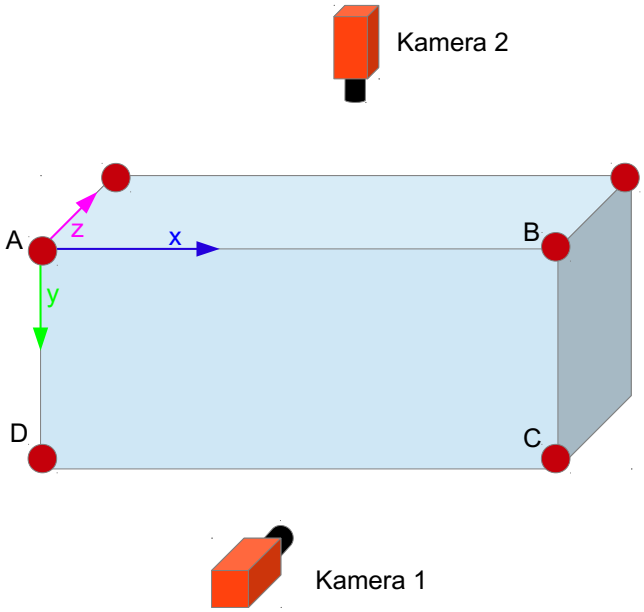


Abbildung 3.7: **Koordinatensystem des Aquariums.** Die Grafik wurde vorab in [87] veröffentlicht.

der Kamera berechnen können. Der Algorithmus wird für jede Kamera separat angewendet und liefert die Kameramatrizen K_1 und K_2. Die Kameramatrizen sind wie folgt definiert:

$$K_{1,2} = (R_{1,2}|\vec{t}_{1,2}) \tag{3.27}$$

mit der Rotationsmatrix $R_{1,2}$ und dem Translationsvektor $\vec{t}_{1,2}$.
Die automatische Erkennung der Aquariumsmarker in der Bildebene wird im folgenden Unterkapitel behandelt.

3.4.1.1 Erkennung der Kalibrierpunkte

Die Kalibrierpunkte werden durch farbige Marker repräsentiert, die mit Hilfe von Farbsegmentierung einfach gefunden und extrahiert werden können. Es stehen zwei verschiedene Markerarten zur Verfügung: zum einen werden rote Golfbälle genutzt, in denen mit Hilfe einer hochpräzisen, computergesteuerten Fräse eine Ecke ausgefräst wird. Die Spitze der Ecke liegt dabei im Zentrum des Golfballs. Es werden sechs Golfbälle auf die Ecken des Aquariums geklebt, so dass diese in jeder Kameraansicht sichtbar sind. Zum anderen werden rote Dreiecke verwendet, die sowohl von oben als auch von vorne an die Aquariumecken geklebt werden (siehe Abbildung 3.8).

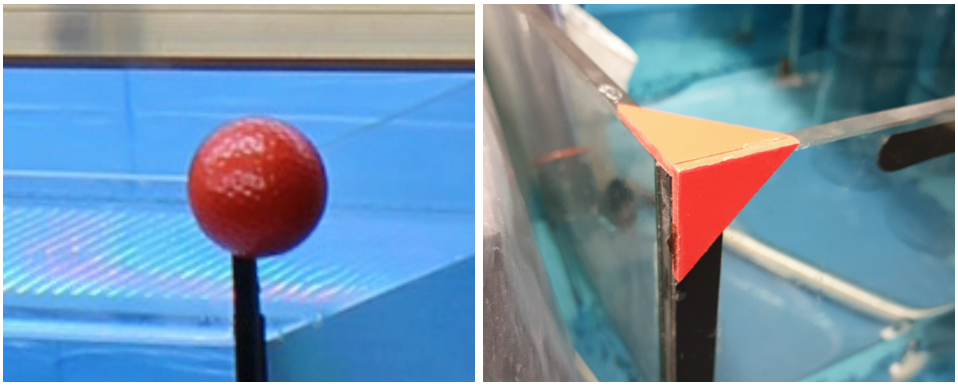


Abbildung 3.8: **Marker am Aquarium.** Zur Markierung der Kalibrierpunkte werden farbige Marker eingesetzt. Es werden zum einen Golfbälle (links) und zum anderen Dreiecke an die Ecken des Aquarium geklebt. Diese können automatisch vom Bildverarbeitungssystem erkannt werden.

Durch die farbliche Unterscheidung der Marker von der restlichen Szene können diese durch eine einfache → Farbsegmentierung extrahiert werden. Zur Bestimmung des Golfball-Mittelpunktes (dieser repräsentiert die Ecke des Aquariums) in der Bildebene wird der Schwerpunkt S des segmentierten Ballfläche mit Hilfe des Bildmoments berechnet (siehe Gleichung 3.13). Im Falle des Dreiecks wird im ersten Schritt die Kontur des Dreiecks extrahiert. Diese wird mit Hilfe des Douglas-Peucker-Algorithmus [41] auf drei Eckpunkte approximiert. Da es sich um ein rechtwinkliges Dreieck handelt und der rechte Winkel (repräsentiert die Ecke des Aquariums) gegenüber der Hypotenuse des Dreiecks liegt, wird die Ecke gegenüber der Hypotenuse als gesuchte Ecke angenommen. Um die Genauigkeit der Eckkoordinate zu erhöhen, wird im zweiten Schritt ein Subpixel-Verfahren angewendet.

3.4.1.2 Assistent zur Ausrichtung der Kameras

Als optimale Kameraausrichtung wird eine parallele Ausrichtung der Bildebene zur Scheibe des Aquariums bzw. zur Wasseroberfläche gesehen. Zudem sollte das Kamerazentrum bei paralleler Ausrichtung der Kamera mittig über dem Zentrum des Aquariums liegen. Um dieses zu gewährleisten, bietet eine Software dem Benutzer vor Beginn des Trackings Hilfestellung bei der Ausrichtung und Platzierung. Anhand der gefundenen Ecken des Aquariums (siehe Unterkapitel 3.4.1.1) wird die Mitte des Aquariums berechnet und in die

Abbildung 3.9: **Screenshot der Kalibriersoftware.** Das obere Bild zeigt die Ansicht der Kamera über dem Aquarium, das untere Bild zeigt die Ansicht der Frontkamera. Der blaue und gelbe Kreis dienen der Kameraausrichtung. Befindet sich der blaue Kreis in der Mitte des Gelben ist die Kamera mittig auf das Aquarium ausgerichtet. Zudem wird dem Benutzer die aktuelle Kameraverkippung angezeigt. Nachdem keine Warnungen mehr angezeigt werden, kann die Kalibrierungsmatrix beider Kameras abgespeichert werden.

Vorschauansicht des Kamerabildes projiziert (blauer Kreis in Abbildung 3.9). Zudem werden die Nick- und Gierwinkel der Kamera im Verhältnis zum Aquarium überprüft und der Benutzer wird im Falle einer Verkippung bzw. Verdrehung der Kamera benachrichtigt. Dabei wird die Verkippung wie folgt durch Indizes approximiert. Die Berechnung wird beispielhaft für Kamera 1 aus Abbildung 3.7 durchgeführt:

Verkippungsindex g um die Gier-Achse

$$g = |\vec{AD}| - |\vec{BC}| \tag{3.28}$$

Verkippungsindex n um die Nick-Achse

$$n = |\vec{AB}| - |\vec{CD}| \tag{3.29}$$

Sollten die Indizes g und n eine untere oder obere Schwelle überschreiten, wird der Benutzer über die Verkippung der Kamera informiert (siehe Abbildung 3.9).

3.5 Brechungsberechnung mittels Ray-Tracing

Zur Berechnung der Lichtbrechung wird im Rahmen dieser Arbeit eine Methode entwickelt, die auf der Lichtstrahlverfolgung (engl. Ray-Tracing) basiert. Dabei wird der Weg des Lichtes vom Objekt bis zum Sensel der Kamera unter Anwendung der physikalischen Gesetze modelliert. Besonders in der Computergrafik findet diese Technik Anwendung, um möglichst realistische Bilder aus 3D-Szenen zu →rendern. Nachteil dieser Technik ist der hohe Rechenaufwand, da für jedes Pixel eine separate Berechnung zur Strahlverfolgung durchgeführt werden muss.
Im Folgenden wird die Berechnung der Brechung für den in Abbildung 3.7 gezeigten Aufbau beschrieben. Bei der Berechnung wird der Strahl entgegengesetzt der physikalischen Richtung vom Kamerazentrum hin zum Objekt verfolgt. Der Lichtstrahl beginnt im Kamerazentrum $S_{1,2}$ und wird an der Luft-Wasser-Grenzebene im Punkt $I_{1,2}$ gebrochen. Die Position des Kamerazentrums $S_{1,2}$ ergibt sich aus den Kameramatrizen $K_{1,2}$ (vgl. Gleichung 3.27).

$$S_{1,2} = -{R_{1,2}}^{-1}\vec{t}_{1,2} \tag{3.30}$$

Ein Strahl $\vec{x}_{1,2}$, welcher im Kamerazentrum startet und im 2D-Punkt X' die Bildebene durchstößt, wird durch die Multiplikation mit der inversen Kamerakalibrierungsmatrix bestimmt. Die Invertierung der Kamerakalibrierungsmatrix wird mit Hilfe der Singulärwertzerlegung durchgeführt.

$$\hat{x}_{1,2} = C_{1,2}^{-1}\vec{x'} \tag{3.31}$$

$\hat{x}_{1,2}$ entspricht dem normierten Richtungsvektor zwischen dem Kamerazentrum S und dem Durchstoßungspunkt I.

$$\hat{x}_{1,2} = \frac{\vec{i}_{1,2} - \vec{s}_{1,2}}{|\vec{i}_{1,2} - \vec{s}_{1,2}|} \tag{3.32}$$

Um den Schnittpunkt $I_{1,2}$ des Strahles $\hat{x}_{1,2}$ mit der Luft-Wasser-Grenzfläche zu berechnen, wird in einem ersten Schritt die Entfernung zwischen Kamerazentrum und Schnittpunkt bestimmt. Diese lässt sich mit Hilfe der Geradengleichung berechnen:

$$\vec{i}_{1,2} = \vec{s}_{1,2} + c_{1,2}\hat{x}_{1,2}. \tag{3.33}$$

Da I_1 (erste Kamera) in der xy-Ebene und I_2 in einer Parallelebene zur xz-Ebene liegt und es nur eine Lösung zur Gleichung 3.33 gibt (Strahl ist nicht parallel zur Ebene), kann $c_{1,2}$ durch folgende Vereinfachung berechnet werden. Da es sich bei $\hat{x}_{1,2}$ um einen Einheitsvektor handelt, gibt $c_{1,2}$ die direkte Entfernung zwischen Kamera und Durchstoßungspunkt an:

$$c_1 = \frac{s_{1z}}{s_{1z} - x_{1z}}. \tag{3.34}$$

Für die Berechnung der Entfernung von der zweiten Kamera hin zum Durchstoßungspunkt mit der Wasseroberfläche muss der Wasserstand w des Aquariums bekannt sein. Dieser wird manuell ermittelt.

$$c_2 = \frac{s_{2y} - w}{s_{2y} - x_{2y}}. \tag{3.35}$$

Mit Hilfe von $c_{1,2}$ und der Gleichung 3.33 kann schlussendlich der Ortsvektor des Durchstoßungspunktes $\vec{i}_{1,2}$ berechnet werden.

Die Lichtbrechung für einen Strahl, der von einem Medium in ein anderes wechselt, kann mit dem Snelliusschen Brechungsgesetz $\frac{\sin\alpha}{\sin\beta} = \frac{n_2}{n_1}$ (vgl. Unterkapitel 3.1.2) berechnet werden. n_1 und n_2 geben dabei die Brechungsindizes der Medien an.
Für Kamera 1 gibt α_1 den Winkel zwischen dem Kamerastrahl x_1 und der Front des Aquariums an; für die zweite Kamera beschreibt α_2 den Winkel zwischen dem Kamerastrahl x_2 und der Wasseroberfläche. Da diese Ebenen parallel zu jeweils einer Ebene des Koordinaten Systems sind, kann die Berechnung vereinfacht werden:

$$\alpha_1 = \sin^{-1}(\frac{x_{1z}}{|\vec{x_1}|}) - \frac{\pi}{2} \tag{3.36}$$

$$\alpha_2 = \sin^{-1}(\frac{x_{2y}}{|\vec{x_2}|}) - \frac{\pi}{2} \tag{3.37}$$

Mit Hilfe des Brechungsgesetzes wird daraus der Winkel β berechnet.

$$\beta_n = \sin^{-1}(\sin\alpha_{1,2}\frac{n_1}{n_2}). \tag{3.38}$$

Wie in Abbildung 3.10 zu sehen, berechnet sich der Winkel γ, um den der Strahl schlussendlich gedreht werden muss, wie folgt:

$$\gamma = \beta - \alpha. \tag{3.39}$$

Der Strahl wird im Punkt $\vec{i}_{1,2}$ um die Achse $\vec{a}_{1,2}$ gedreht. $\vec{a}_{1,2}$ ist orthogonal zur Ebene, die durch den Vektor $\vec{x}_{1,2}$ und den zentralen Kamerastrahlen $\vec{c}_{1,2}$ aufgespannt wird. c' beschreibt das zentrale Pixel auf der Bildebene.

$$\vec{a}_{1,2} = ({C_{1,2}}^{-1}c') \times \vec{x}_{1,2} \tag{3.40}$$

Zur Drehung des Strahls wird die Rotationsmatrix $R_{a_{1,2}}$ mit Hilfe des norma-

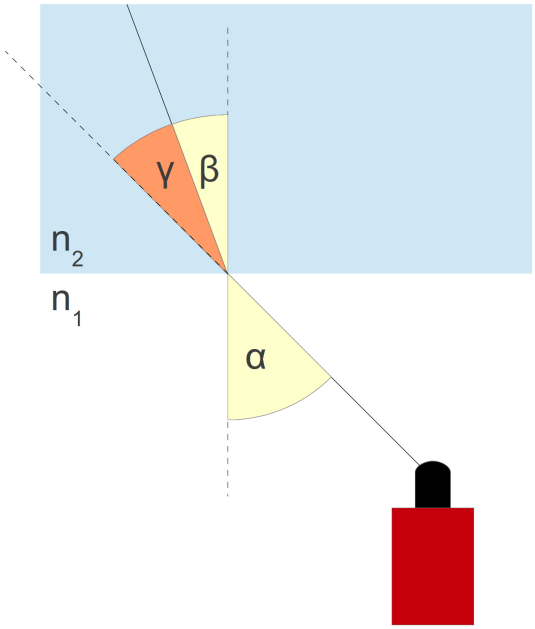


Abbildung 3.10: **Brechung des Strahles.** γ beschreibt den Winkel der Brechung. Diese Grafik wurde vorab in [87] veröffentlicht.

lisierten Achsvektors $\hat{a}_{1,2}$ berechnet.

$$R_{a_{1,2}} = \begin{pmatrix} d_1 & d_2 & d_3 \\ e_1 & e_2 & e_3 \\ f_1 & f_2 & f_3 \end{pmatrix} \quad \text{mit}$$

$$\begin{aligned} d_1 &= \hat{a}_x^2(1-\cos\gamma) + \cos\gamma \\ d_2 &= \hat{a}_y\hat{a}_x(1-\cos\gamma) + \hat{a}_z\sin\gamma \\ d_3 &= \hat{a}_z\hat{a}_x(1-\cos\gamma) - \hat{a}_y\sin\gamma \\ e_1 &= \hat{a}_x\hat{a}_y(1-\cos\gamma) - \hat{a}_z\sin\gamma \\ e_2 &= \hat{a}_y^2(1-\cos\gamma) + \cos\gamma \\ e_3 &= \hat{a}_z\hat{a}_y(1-\cos\gamma) + \hat{a}_x\sin\gamma \\ f_1 &= \hat{a}_x\hat{a}_z(1-\cos\gamma) + \hat{a}_y\sin\gamma \\ f_2 &= \hat{a}_y\hat{a}_z(1-\cos\gamma) - \hat{a}_x\sin\gamma \\ f_3 &= \hat{a}_z^2(1-\cos\gamma) + \cos\gamma \end{aligned} \tag{3.41}$$

Der gebrochene Strahl $\vec{x}_{r_{1,2}}$ ergibt sich aus

$$\vec{x}_{r_{1,2}} = R_{a_{1,2}}\hat{x}_{1,2} \tag{3.42}$$

und beginnt im Punkt $\vec{i}_{1,2}$.

In der Berechnung wurde die Brechung, die zusätzlich durch das Glas des Aquariums entsteht vernachlässigt. Da sich die Brechungsindices von Glas und Wasser relativ zum Brechungsindex von Luft wenig unterscheiden, ist diese Vereinfachung möglich. Sie könnte jedoch auf dieselbe Weise berechnet werden.

3.6 Detektion

Die Detektion der Fische erfolgt in einer definierten Umgebung, mit statischen Kameras, gleichbleibender Belichtung und wenige Störfaktoren. Dies sind gute Voraussetzungen für die Detektion der Fische. Bedingt durch den Einsatzzweck des Systems sind jedoch einige Schwierigkeiten und besonderen Anforderungen zu nennen. Wie in Kapitel 3.2 geschildert ergeben sich folgende Punkte:

- **hoher Automatisierungsgrad:** Da der Experimtator während eines Versuches durch das Experiment ausgelastet ist, soll die Detektion der Fische vollautomatisch durchgeführt werden.
- **Echtzeitfähigkeit:** Da das System für die spätere interaktive Steuerung in Echtzeit arbeiten muss, muss auch die Detektion diesen Ansprüchen gerecht werden. Für die 3D-Berechnung werden zwei Kameras eingesetzt. Um Echtzeitfähigkeit zu erreichen, muss der Detektionsalgorithmus auf handelsüblicher Hardware zwei Kamerabilder in der Zeit eines Frames bearbeiten können.
- **hohe Präzision:** Um im späteren Verlauf auch die Biegung und Ausrichtung der Fische extrahieren zu können, ist eine präzise Extraktion erforderlich.

Eine gängige Methode zur Detektion von bewegten Objekten in statischer Umgebung sind Hintergrundsubtraktionsverfahren (vgl. Kapitel 3.1.4.1). Auch zur Detektion von Fischen wird dieses Verfahren häufig angewendet (vgl. [36]). Ein großer Vorteil des Verfahrens ist die hohe Präzision, die eine pixelgenaue Segmentierung des Vordergrundobjektes ermöglicht. Auch in dieser Arbeit wurde ein auf Hintergrundsegmentierung basiertes Verfahren entwickelt. Um einen hohen Automatisierungsgrad der Anwendung zu erreichen, werden zwei

verschiedene Verfahren kombiniert und die Vorteile jedes einzelnen genutzt. Dabei handelt es sich zum einen um das auf Gauß-Mischverteilungen basierende Gaussian-Mixture-Models Subtraktionsverfahren (**GMS**) [159]. Dieses ist sehr rechenaufwendig und wird nicht für Echtzeitanwendungen empfohlen (vgl. [9]). Das Verfahren liefert jedoch gute Ergebnisse bei der Detektion von bewegten Objekten trotz kurzer Initialisierung.
Da der Hintergrund während der Ausführung durchgehend aktualisiert wird, kann es jedoch passieren, dass Vordergrundobjekte, die sich längere Zeit nicht bewegen, in den Hintergrund übergehen. Dieser Effekt wurde auch bei Tests mit Fischen festgestellt: Fische, die länger auf einer Stelle verharrten, wurden nur noch teilweise detektiert (siehe Abbildung 3.11). Im hier beschriebenen Projekt wird das GMS-Verfahren zur Initialisierung des Hintergrundes eines zweiten Verfahrens genutzt. Bei diesem handelt es sich um das Codebook-Hintergrundsubtraktionsverfahren (**CB**) [60], welches ein sehr gutes Segmentierungsergebnis liefert und auf handelsüblicher Hardware in Echtzeit ausgeführt werden kann (vgl. [9]). Eine Beschreibung des Verfahrens ist in Kapitel 3.1.4.1 zu finden. Das Verfahren bildet die Pixel des Hintergrundes in sogenannten Codewörtern ab. Wenn der Hintergrund statisch ist, wird jedes Pixel lediglich durch ein Codewort abgebildet, welches das Sensorrauschen beinhaltet. Falls sich der Hintergrund wiederholend verändert (im Fall des Aquariums könnten z. B. die Blätter eine Wasserpflanze wiederkehrend ins Bild treiben), wird für jede sich wiederholende Änderung ein neues Codewort abgespeichert. Der Hintergrund des hier verwendeten CB-Verfahrens wird anders als das im ersten Schritt verwendete GMS-Verfahren nicht durchgängig aktualisiert, sondern ausschließlich während einer Initialisierungsphase zu Beginn erzeugt. Im besten Fall sind während der Generierung des Hintergrundes keine Vordergrundobjekte im Bild zu erkennen. Dies lässt sich im Fall des mit Fischen bestückten Aquariums nicht einhalten. Aus diesem Grund werden die zur Initialisierung verwendeten Bilder mit Hilfe des GMS-Verfahrens vorverarbeitet:

1. Der Suchraum wird auf Basis der während der Kalibrierung gefundenen Ecken des Aquariums begrenzt.

2. Das GMS-Verfahren liefert eine ungenaue Kontur aller sich im Aquarium befindlichen beweglichen Objekte. Auch ein Teilbereich stillstehender Fi-

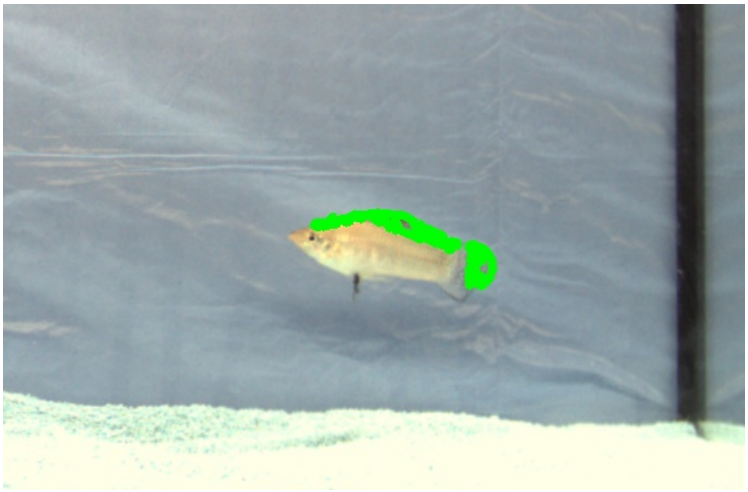


Abbildung 3.11: **Geschrumpfter Vordergrund.** Ein Großteil des zuvor segmentierten Fisches ist auf Grund von Stillstand in den Hintergrund übergegangen. Lediglich die Flossen, die sich durchgehend bewegen, sind noch im Vordergrund abgebildet (grün umrandet). Diese Grafik wurde vorab in [87] veröffentlicht.

sche wird erkannt: Tests zeigten, dass stillstehende Breitflossenkärpflinge ihre Rücken- und Schwanzflossen kontinuierlich bewegen. Diese werden vom GMS-Verfahren, wie in Abbildung 3.11 zu sehen, detektiert.

3. Um Vordergrundobjekte nicht mit in die Hintergrundgenerierung des CB-Verfahrens zu übernehmen, werden die Bereiche in der Initialisierung des CB-Verfahrens ausgespart. Dazu wird je eine → Boundigbox um die zuvor gefundenen Konturen gelegt. Um auch bei einer Teildetektion (es wurden z. B. nur Flossen erkannt) des Fisches den ganzen Fisch auszuschließen, wird ein Sicherheitsbereich um die Boundingbox gelegt. Dieser hat eine feste Breite und Höhe, welche manuell entsprechend der Fischart und des Kameraabstandes definiert wird (siehe Abbildung 3.12).

4. Im hier beschriebenen Anwendungsfall hat sich in Kombination mit dem CB-Verfahren besonders das → YCbCr-Farbmodell bewährt. Dieses teilt die Farbinformationen in die Grundhelligkeit (Y) und in zwei Farbkomponenten (Cb - blau-gelb und Cr - rot-grün → Chrominanz) auf. Dies ermöglicht eine getrennte Behandlung von Farbe und Helligkeit im weiteren Verlauf des Verfahrens. Die Bilder der Kameras werden vor der

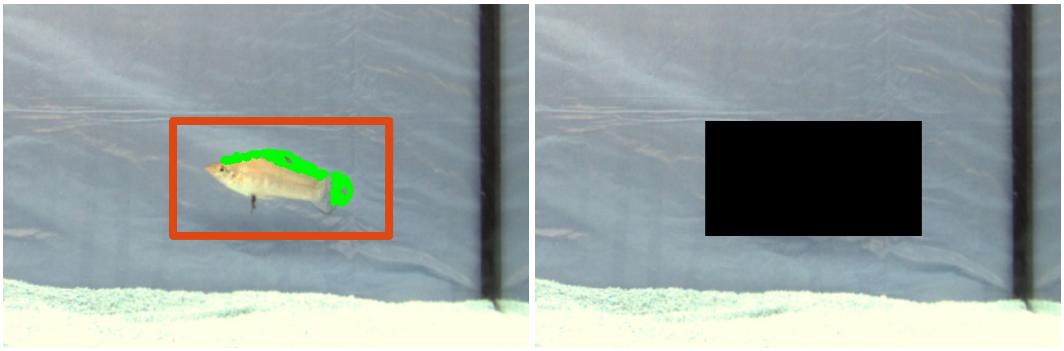


Abbildung 3.12: **Partielle Initialisierung des Hintergrundes.** Es wird ein Sicherheitsbereich um die vom GMS-Verfahren gefundenen Konturen gelegt (rotes Rechteck). In der Initialisierung wird dieser Bereich ausgespart (rechts).

Weiterverarbeitung in dieses Farbmodell umgerechnet.

5. Bei der Initialisierung des CB-Verfahren werden die durch die zuvor erstellte →Maske (Bildverarbeitung) abgedeckten Bereiche ausgespart. Um sicherzustellen, dass alle Bereiche ausreichend initialisiert wurden, wird die Anzahl der Initialisierungen pro Pixel gezählt. Sobald alle Pixel k-mal initialisiert wurden, wird die Initialisierung abgeschlossen. In der Praxis hat sich $k = 50$ bewährt.

6. Während der Vordergrunddetektion ist nur des CB-Verfahren mit statischem Hintergrund aktiv und segmentiert die Vordergrundobjekte. Aus den gefundenen Vordergrund-Segmenten werden mit dem Kontur-Such-Verfahren von Suzuki [135] die Konturen A_{1n} (Kamera 1) und A_{2n} (Kamera 2) extrahiert.

3.6.1 Kontur-Referenzierung

Zur Erzeugung einer 3D-Repräsentation der Fische werden Kontur-Paare (A_{1n}, A_{2n}) beider Kameras benötigt, die dasselbe Objekt beschreiben. Da gegebenenfalls mehrere Konturen pro Kamera gefunden werden können, muss eine eindeutige Zuordnung der Konturen zum Objekt erfolgen. Diese lässt sich mit Hilfe der Kamerageometrie und der Brechungsberechnung durchführen: da der Pfad eines jeden Pixel-Strahles bekannt ist (siehe Kapitel 3.5), kann der Abstand zweier Strahlen $\vec{x_{r_1}}$ (Kamera 1) und $\vec{x_{r_2}}$ (Kamera 2) zweier potenziell

verbundener Konturen A_{1n} und A_{2n} berechnet werden. Um den Berechnungsaufwand möglichst gering zu halten, wird nur ein Strahl pro Kontur getestet. Dabei muss sichergestellt werden, dass die Strahlen der beiden Konturen jeweils denselben 3D-Objekt-Punkt repräsentieren. Im hier beschriebenen Fall wird dies über die gemeinsame x-Achse (Abbildung 3.7) realisiert. Für jede Kontur wird das Pixel x'_1 bzw. x'_2 gesucht, das den kleinsten x-Wert hat. Anschließend werden mit Hilfe von Gleichung 3.42 die Strahlen $\vec{x'_1}$ und $\vec{x'_2}$ berechnet. Der Abstand a der Strahlen berechnet sich unter der Annahme, dass diese nicht parallel sind, wie folgt.

$$a = \frac{|(\vec{i}_1 - \vec{i}_2) * (\vec{x'_1} \otimes \vec{x'_2})|}{|\vec{x'_1} \otimes \vec{x'_2}|} \tag{3.43}$$

$\vec{i}_{1,2}$ beschreiben dabei die Durchstoßungspunkte der Strahlen mit der Aquariumscheibe bzw. der Wasseroberfläche. Im perfekten Fall schneiden sich die Geraden im Objekt und der Abstand ist $a = 0$. Da es jedoch systembedingt zu Ungenauigkeiten kommt (z.B. $\rightarrow$ Rasterung) ist ein Schnittpunkt zwischen den Geraden unwahrscheinlich. Aus diesem Grund werden zwei Konturen aus unterschiedlichen Ansichten einem Objekt zugeteilt, sobald $a < \epsilon$ ist (mit einer festen Schwelle ϵ).

3.6.2 Spiegelung

An den Seitenwenden des Aquariums können Spiegelungen auftreten (siehe Abbildung 3.13), die ebenfalls durch das CB-Verfahren detektiert werden. Um die gespiegelten Konturen auszuschließen, werden zwei verschiedene Strategien angewendet. Zum einen können über die in Unterkapitel 3.6.1 beschriebene Konturreferenzierung die seitlichen Spiegelungen ausgeschlossen werden. Zum anderen kann überprüft werden, ob die Schnittpunkte (bzw. Punkte kürzester Entfernung) der Strahlen $\vec{x'_1}$ und $\vec{x'_2}$ innerhalb oder außerhalb des Aquariums liegen. Dazu wird die folgende Gleichung mit Hilfe eines Gleichungssystems nach r und s gelöst:

$$\vec{i}_1 + s\vec{x'_1} - \vec{i}_2 + r\vec{x'_2} = \vec{x'_1} \otimes \vec{x'_2}. \tag{3.44}$$

Daraus ergeben sich die Ortsvektoren $\vec{k_1}$ und $\vec{k_2}$ auf den Geraden mit dem

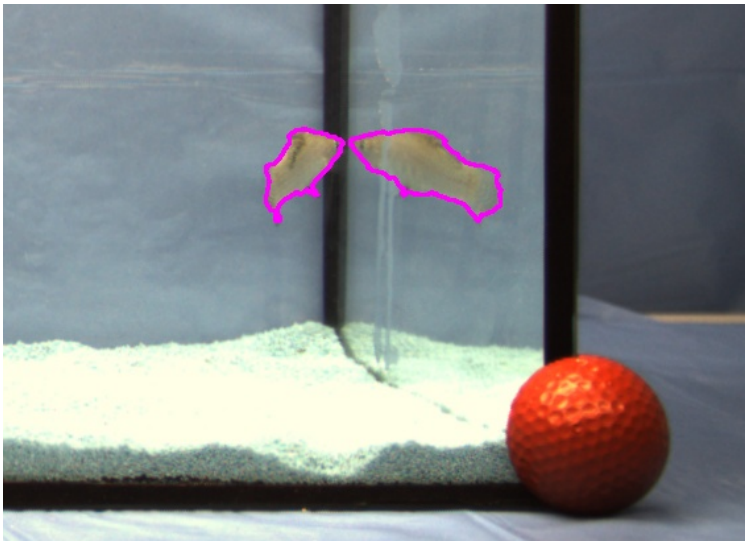


Abbildung 3.13: **Gespiegelter Fisch.** Im Aquarium ergeben sich Spiegelungen sobald sich der Fisch den Seitenscheiben des Aquariums nähert. Die linke Kontur zeigt den Fisch, die rechte das Spiegelbild. Diese Grafik wurde vorab in [87] veröffentlicht.

geringsten Abstand zueinander:

$$\vec{k_1} = \vec{i_1} + s\vec{x_1'} \tag{3.45}$$

$$\vec{k_2} = \vec{i_2} + r\vec{x_2'}. \tag{3.46}$$

Abschließend wird geprüft, ob $\frac{(\vec{k_2}+\vec{k_2})}{2}$ innerhalb der Aquariums liegt. Falls es außerhalb liegt, wird das Kontur-Paar verworfen.

3.6.3 Schatten

Das hier verwendete CB-Verfahren detektiert bereits kleine Änderungen im Bild. Trotz guter Ausleuchtung des Aquariums werfen Fische leichte Schatten, die detektiert und dem Vordergrund hinzugefügt werden (siehe Abbildung 3.14). Jedes Pixel-Codewort beschreibt den Hintergrund im → YCbCr-Farbraum. In diesem Farbraum ist die → Luminanz (Helligkeit) von der → Chrominanz (Farbigkeit) separiert. Weicht der Pixelwert des aufgenommenen Bildes in nur einem der Farb- oder Helligkeitskanäle ab, wird es als Vordergrund detektiert. Da sich bei einer Abschattung hauptsächlich die Luminanz verändert,

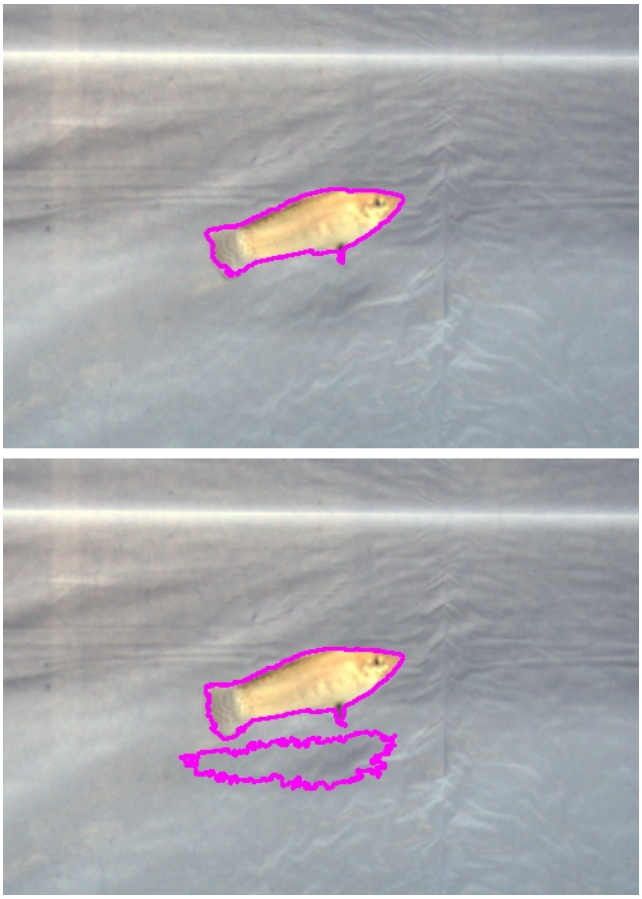


Abbildung 3.14: **Schatten des Fisches.** Das untere Bild zeigt den segmentierten Fisch und dessen Schatten. Im oberen Bild wurde der Schatten erkannt und die Kontur des Schattens entfernt. Diese Grafik wurde vorab in [87] veröffentlicht.

kann der Grenzwert für den Helligkeitskanal erweitert werden, um somit Veränderungen, die durch Schatten hervorgerufen werden, nicht zu detektieren.

3.6.4 Bestimmung der Fischausrichtung

Die Fischausrichtung wird über die Gerade, die vom geschätzten Mittelpunkt des Fisches zum Fischmaul verläuft definiert. Dazu muss zum einen das Fischmaul als auch der Mittelpunkt des Fisches in der Kontur gefunden werden. Als Mittelpunkt des Fisches wird die Stelle in der Fischkontur angenommen, in der der Mittelpunkt des größten Kreises innerhalb der Kontur liegt. Dieser

Punkt liegt im Falle der Breitflossenkärpflinge im vorderen Teil des Fisches (siehe Abbildung 3.15). Zur Bestimmung dieser Punkte wird im ersten Schritt ein Kreis mit maximalem Radius in die Kontur gelegt. Zur Bestimmung der Kreisposition und des Kreisradius wird das Pixel gesucht, dass innerhalb der Kontur liegt und den größten Abstand zu einem der Konturpixel hat. Der Abstand wird als Radius des Kreises angenommen und die Position des Pixels als Mittelpunkt M des Kreises. Im folgenden Schritt wird der Kreis orthogonal zur Längsrichtung des Fisches halbiert. Dazu wird das Konturpixel, welches den geringsten Abstand zum Kreismittelpunkt hat, gesucht. Dieses liegt auf dem Kreis. Der Kreis wird durch die verlängerte Gerade zwischen Kreismittelpunkt und nächstem Konturpixel geteilt (siehe Abbildung 3.15). Auch die Kontur wird an diesen Stellen in zwei Teilkonturen geteilt. Im letzten Schritt wird für jede Teilkontur der Konturpunkt gesucht, der am weitesten vom Mittelpunkt des Kreises entfernt ist. Dies sind in den meisten Fällen das Fischmaul und die Schwanzspitze. Da das Maul auf Grund der Annahme, dass sich der Kreis im vorderen Teil des Fisches befindet, näher am Mittelpunkt liegt, wird das Pixel mit der kürzeren Entfernung als Maul P_m angenommen. Der Winkel der Fischausrichtung a wird wie folgt und in Bezug zur positiven X-Achse der Bildebene gegen den Uhrzeigersinn gemessen:

$$a = \begin{cases} atan(\frac{P_{my}-M_y}{P_{mx}-M_x})\frac{180}{\pi}, & P_{mx}-M_x > 0 \\ atan(\frac{P_{my}-M_y}{P_{mx}-M_x})\frac{180}{\pi}+180, & P_{mx}-M_x < 0 \\ atan(\frac{P_{my}-M_y}{P_{mx}-M_x})\frac{180}{\pi}+360, & P_{mx}-M_x > 0 \wedge P_{my}-M_y < 0 \end{cases} \tag{3.47}$$

3.7 Vereinigung der 2D-Konturen zum 3D-Modell

Bei klassischen Stereokamera-Aufbauten sind die Kameras parallel zueinander ausgerichtet und Objekte werden aus einem sehr ähnlichen Blickwinkel betrachten. Basierend auf der → Epipolargeometrie, werden → Merkmalspunkt (Bildverarbeitung)-Paare in beiden Ansichten gesucht und ein Tiefenwert über eine Triangulation bestimmt. Im hier beschriebenen Fall sind die Kameras orthogonal zueinander ausgerichtet. Im Falle des Fisches bringt dies den Vorteil, dass sowohl die Biegung des Fisches von oben als auch die Seitenflan-

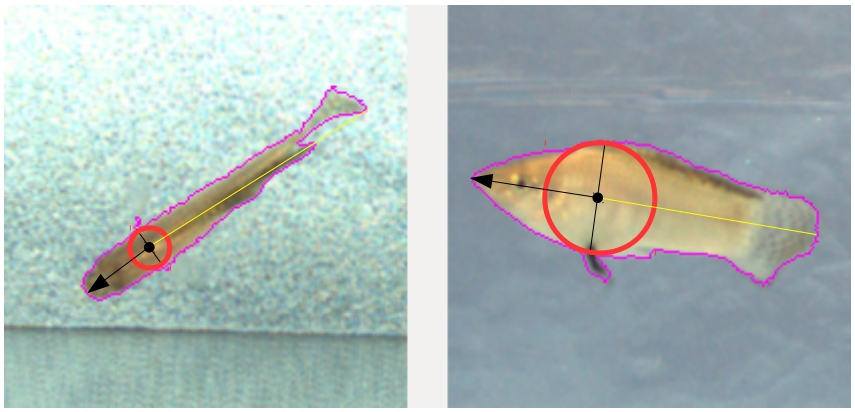


Abbildung 3.15: **Bestimmung des Mauls.** Auf Basis der Kontur (lila) wird die Position des Fischmauls (Pfeilspitze) bestimmt. Die eingezeichneten Kreise sind die jeweils größtmöglichen innerhalb der Kontur. Die Ausrichtung des Fisches in der jeweiligen Ansicht gibt der Pfeil an.

ke beobachten werden kann. Dadurch ist es jedoch nicht möglich Textur-basierte Merkmalspunkt-Korrespondenzen zu finden. Darum wird in diesem Abschnitt ein Verfahren angewendet, welches die vorhandenen Informationen (2D-Konturen verschiedener Ansichten) in ein 3D-Kastenmodell des Fisches überführt, welches sowohl die für den weiteren Verlauf wichtige Fischbiegung als auch die genaue 3D-Pose beinhaltet.
Grundprinzip des vorgestellten Verfahrens ist die Berechnung des Schnittkörpers beider extrudierter ($\rightarrow$ Extrusion) Konturen. Im ersten Schritt wird die Kontur der ersten Kamera entlang der Kamerahauptachse extrudiert. Dazu werden Dreiecke der Fischkontur-Strahlen der ersten Kamera $\vec{x}_{1_n}$ erzeugt, die von der Front- bis zur Rückscheibe des Aquariums reichen. n beschreibt dabei die Nummer des Konturpixels. $\vec{x}_{1_n}$ und $\vec{x}_{1_{n+1}}$ sind Strahlen benachbarter Konturpixel. I_{1_n} ist der Durchstoßungspunkt des Strahles an der Frontscheibe $\vec{x}_{1_n}$ und gleichzeitig der Startpunkt dessen. Da das 3D-Objekt innerhalb des Aquariums liegt, wird ebenfalls ein Endpunkt E_n des Strahls $\vec{x}_{1_n}$ definiert (siehe Abbildung 3.16).

$$E_n = \hat{x}_{1_n} l + I_{1_n} \tag{3.48}$$

l beschreibt die maximale Länge eines Strahles innerhalb des Aquariums.

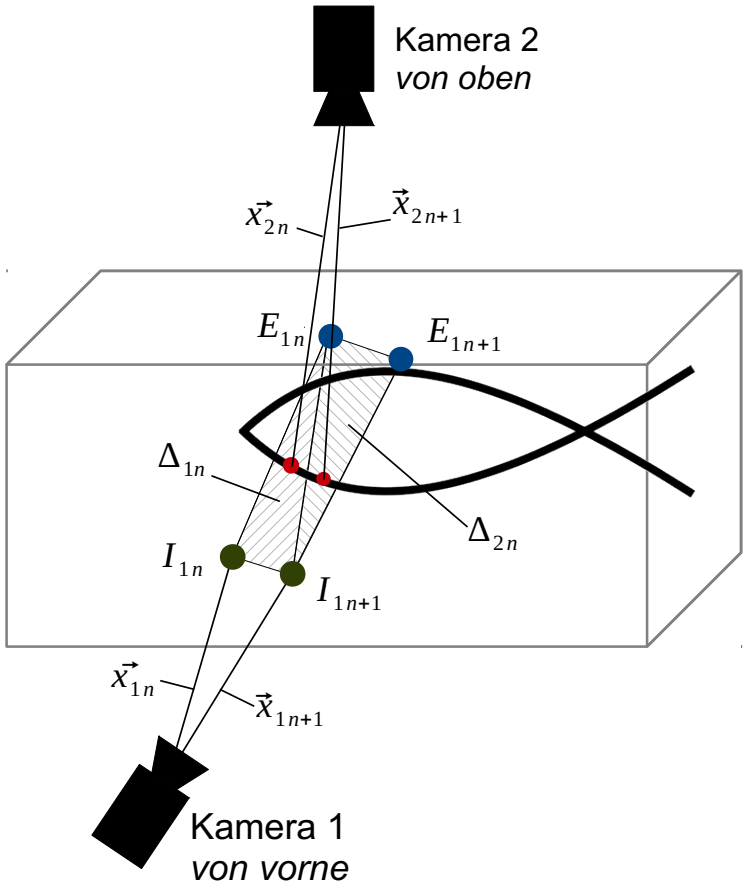


Abbildung 3.16: **Erzeugung des 3D-Kastenmodells.** Das Bild zeigt exemplarisch zwei Dreiecke $\triangle_{1n}$ und $\triangle_{2n}$, welche durch die Punkte I_{1_n}, $I_{1_{n+1}}$ (grün) und E_n und E_{n+1} (blau) aufgespannt werden. Die Schnittpunkte der Konturstrahlen aus Kamera 2 mit dem Mesh sind im Bild rot gekennzeichnet.

Die Dreiecke entlang der Kontur sind wie folgt definiert:

$$\begin{aligned} \triangle_{1n} &= \triangle(I_{1_n}, E_n, I_{1_{n+1}}) \\ \triangle_{2n} &= \triangle(E_n, I_{1_{n+1}}, E_{n+1}) \end{aligned} \tag{3.49}$$

$\triangle_{1n}$ und $\triangle_{2n}$ bilden ein Dreieck-Mesh($\rightarrow$ Mesh (Polygon)), welches die extruierte (gebrochene) Kontur der ersten Kamera durch das Aquarium hindurch abbildet (siehe Abbildung 3.16). Um das 3D-Kastenmodell zu erhalten, werden im zweiten Schritt die Durchstoßungspunkte der Konturstrahlen der zweiten Kamera $\vec{x}_{2_n}$ mit dem Dreieck-Mesh berechnet. Zur Berechnung des Durchstoßungspunktes wird das von Möller und Trumbore entwickelte Verfahren (*fast, minimum storage ray-triangle intersection*) [78] zur schnellen Berechnung von

Schnittpunkten zwischen Strahlen und Dreiecken benutzt. Alle Schnittpunkte zwischen Mesh und Konturstrahlen werden abgespeichert. Die gefundenen Punkte werden schlussendlich zur Generierung des 3D-Kasten-Mesh-Modells genutzt.

3.8 Tracking

Ziel des Trackings ist es, die Fische nach der Detektion (vgl. Kapitel 3.6) über den weiteren Verlauf der Videosequenz zu verfolgen. In der hier beschriebenen Arbeit wird das Tracking auf zwei unterschiedlichen Ebenen durchgeführt: zum einen wird die Kontur und zum anderen die 3D-Position getrackt.
Die Konturen der Fische sind für die Erzeugung der 3D-Kastenmodelle (vgl. Kapitel 3.7) und zur Bewegungsrekonstruktion (vgl. Kapitel 5) von Bedeutung. In der Praxis hat sich gezeigt, dass der manuelle Eingriff beim Einfangen und Freilassen des Testfisches mit einem Plexiglaszylinder (siehe Unterkapitel 3.2.1) Wellen auf der Wasseroberfläche und damit eine ungeordnete Lichtbrechung verursacht. Diese hat zur Folge, dass bei der Hintergrundsubtraktion Fehldetektionen auftreten und eine fehlerfreie Verfolgung der Kontur nicht gewährleistet werden kann (siehe auch Abbildung 3.17). Aus diesem Grund wurde ein zweites Verfahren entwickelt, welches auch unter den oben genannten Bedingungen zuverlässig funktioniert. Anders als bei dem Kontur-Tracking-Verfahren ist mit diesem lediglich die Verfolgung der 3D-Position des Fisches möglich.

3.8.1 Kontur-Tracking

Das hier vorgestellte Kontur-Tracking basiert auf dem in Kapitel 3.1.4.3 beschriebenen *tracking-by-detection*-Ansatz und vereint diesen mit einer Methode des Punkt-Trackings. Der *tracking-by-detection*-Ansatz nutzt zur Verfolgung eine sich wiederholende Detektion des Objektes und bringt einige Einschränkungen und Voraussetzungen mit sich:

1. **Verfolgung von Individuen:** Da in jedem Frame eine neue Detektion durchgeführt wird, ist eine Objektzuordnung von Frame zu Frame nicht beinhaltet und die Gefahr einer Verwechslung von Individuen steigt. Im

hier beschriebenen Anwendungsfall ist die Gefahr einer Verwechslung jedoch stark reduziert, da die Objekte von zwei Ansichten (Top- und Frontaufnahme) beobachtet werden und es somit selbst bei Verdeckung in einer Ansicht, eine weitere Ansicht zur Verfügung steht. Des Weiteren verfügen die Kameras über eine hohe Framerate (vgl. Kapitel 2.2.1), die auf Basis der Fischbewegungsgeschwindigkeit ausgewählt wurde. Somit ist selbst bei hoher Bewegungsgeschwindigkeit eine hohe Überdeckung der Silhouetten zweier aufeinander folgender Frames gewährleistet, was die Zuordnung vereinfacht.

2. **Performance:** Eine Detektion ist oft rechenaufwendiger als der Einsatz von Tracking-Algorithmen. In diesem Projekt wurde bewusst eine schnelle Detektionsmethode gewählt, die auch für Echtzeitanwendungen geeignet ist.

3. **Stabilität:** In Falle von Objektverdeckungen kann es zum Verlust des Objektes im Trackingsystem kommen. Wie im ersten Punkt genannt, kann dies durch die Aufnahme aus einer weiteren Ansicht vermieden werden.

Zur Verfolgung der Fische über den Verlauf der Bildfolge hinweg müssen die Konturen von Frame zu Frame verfolgt und referenziert werden. Im hier beschriebenen Projekt handelt es sich bei den Konturen nicht um einzelne, sondern um Konturpaare (A_{1n}, A_{2n}, vgl. Kapitel 3.6.1). Zum Abgleich der Konturpaare wird von Frame m zu Frame $m+1$ die Summe d der euklidischen Abstände der Konturschwerpunkte S_{1n} und S_{2n} (vgl. Gleichung 3.13) berechnet.

$$d = |\overrightarrow{S_{1n_m} S_{1n_{m+1}}}| + |\overrightarrow{S_{2n_m} S_{2n_{m+1}}}| \tag{3.50}$$

Es wird angenommen, dass die Konturpaare mit der kleinsten Abweichung d dasselbe Objekt n in aufeinanderfolgenden Frames beschreiben.

3.8.2 3D-Positionstracking unter Berücksichtigung von Verdeckung und ungeordneter Lichtbrechung durch Wasserwellen

Der Ablauf von Partnerwahl-Experimenten mit Fischen folgt festen Regeln (vgl. Unterkapitel 3.2.1). Soll das Tracking-System während eines Versuches benutzt werden, darf dies keine Veränderungen für das eigentliche biologische Experiment mit sich bringen. Wie bereits beschrieben kommt es beim Einsatz des Kontur-Tracking-Systems während des Experimentes zu einem Problem: durch das Einfangen des Fisches während des Experimentes durch den Experimentator bilden sich auf der Wasseroberfläche Wasserwellen, welche die unterliegenden Strukturen durch ungeordnete Lichtbrechung verschwimmen lassen. Dies hat besonders bei der Hintergrundsubtraktion schwerwiegende Folgen: Teile des Hintergrundes werden dem Vordergrund zugeordnet und es bilden sich großflächige Segmente, die eine kontinuierliche Verfolgung der Fische unterbrechen. Versuche haben gezeigt, dass dieser Effekt mehr als 20 Sekunden nach der Bewegung des Plexiglaszylinders andauern kann (siehe Abbildung 3.17). Dieser Prozess findet jedoch nur zu Beginn eines jeden Experimentes statt, in dem der Fisch eingefangen wird. Speziell für diese Phase wurde ein automatisches, echtzeitfähiges 3D-Tracking-System entwickelt, welches die Position im Dreidimensionalen verfolgen kann.

Der Effekt der ungeordneten Lichtbrechung beeinflusst viele Verfahren, die auf der Basis von Objektbewegung eine Verfolgung realisieren; wie z. B. auch Verfahren, die auf den → Optischen Fluss setzen. Aus diesem Grund wird hier das weit verbreitete CAMShift-Tracking (siehe Kapitel 3.1.4.4) eingesetzt, welches auf Rückprojektion des Objekt-Farbhistogramms beruht. Dieses wird separat auf beide Videostreams angewendet. Die so gewonnenen 2D-Positionen des Fisches werden in einem weiteren Schritt zu einer 3D-Position vereint. Da eines der Projektziele die Erstellung eines vollautomatischen Tracking-Systems ist, umfasst die vorgestellte Methode eine automatische Initialisierung des CAMShift-Verfahren. Zur Überbrückung von Fischverdeckungen durch den Experimentator, wird die Position und Bewegung zudem über einen Kalman-Filter gefiltert und geschätzt. Sollte es trotzdem zu einem kompletten Verlust des Fisches kommen, verfügt das Verfahren zudem über einen Mechanismus

zur Re-Initialisierung.

3.8.2.1 Automatische Initialisierung

Basis des hier verwendeten CAMShift-Trackers ist die Rückprojektion des Objekt-Farbhistogramms (siehe Abbildung 3.5a) auf das ankommende Bild. Um ein Farbhistogramm des Objektes zu erhalten, wird in vielen Fällen das Objekt manuell ausgewählt. Um dem Projektziel eines automatischen Systems gerecht zu werden, wird dieser Schritt automatisiert. Dazu dient die in Kapitel 3.6 vorgestellte Hintergrund-Subtraktionsmethode. Ihre Anwendung ist während der Akklimatisierungsphase möglich, da in dieser Phase kein Eingriff in das Becken nötig ist und die Wasseroberfläche ruhig ist (siehe auch Abbildung 3.18). Auch die Dauer dieser Phase von 5 bis 10 Minuten ist zur Initialisierung der Hintergrund-Subtraktionsmethode und des CAMshift-Verfahrens ausreichend.
Die mittels dieser Methode extrahierten Konturpaare (A_{1n}, A_{2n}) dienen als Maske der Histogrammberechnung. Alle Pixel innerhalb der Konturen werden zur Berechnung genutzt. Neben dem Farbhistogramm benötigt das CAMShift-Verfahren die initiale Position des Fisches in der Bildebene sowie die Größe in Pixelkoordinaten. Auch diese Informationen lassen sich aus der Kontur ableiten. Dazu wird eine $\rightarrow$ Boundigbox um die Kontur gelegt. Ihre Position und Ausdehnung dienen als Initialisierungswerte des CAMshift-Verfahrens.
Im weiteren Verlauf des Verfahrens wird nach Ankunft eines jeden neuen Bildes eine Rückprojektion des Histogramms durchgeführt. Um die Rechenzeit dieses Schrittes so klein wie möglich zu halten, wird dieser Prozess lediglich im Bereich der letzten Fundstelle des Fisches ausgeführt, in dem sich dieser unter Berücksichtigung seiner maximalen Geschwindigkeit befinden kann. Daraufhin verschiebt der Algorithmus das Suchfenster in Richtung der höchsten Dichte der Rückprojektion und passt die Größe des Fensters an. Der genaue Ablauf des Verfahrens ist in Kapitel 3.1.4.4 zu finden. Als Ergebnis liefert der Algorithmus den Mittelpunkt des Suchfensters als Pixelkoordinate ${p'}_1$ (Frontkamera) und ${p'}_2$ (Topkamera) sowie ihre Ausrichtung und Ausdehnung.

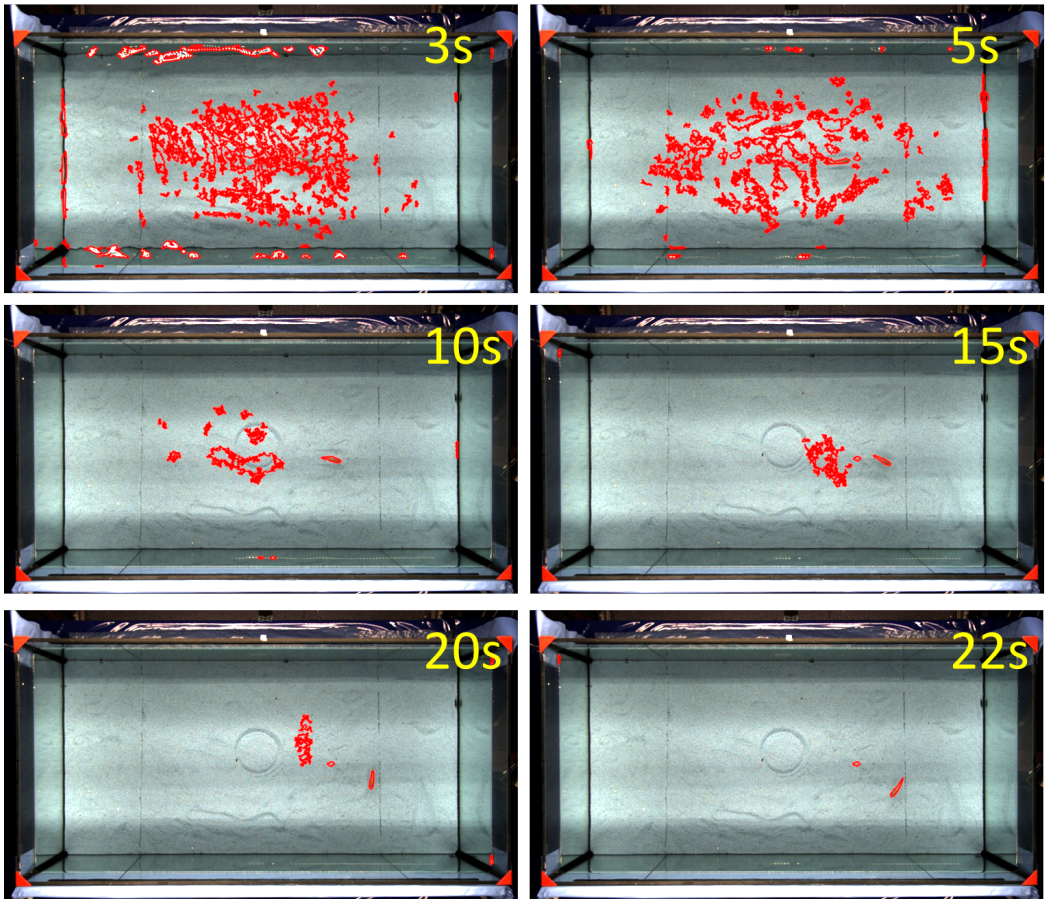


Abbildung 3.17: **Fehldetektionen der Hintergrundsubtraktion bei welliger Wasseroberfläche (Ansicht von oben).** Bei welliger Wasseroberfläche kommt es zur ungeordneten Lichtbrechung und die Hintergrundsubtraktion erkennt Teile des Hintergrundes als Vordergrundobjekte. Nach einem manuellen Eingriff des Experimentators dauert es mehr als 20 Sekunden bis sich die Wasseroberfläche beruhigt hat und keine Fehldetektionen mehr auftreten. Diese Grafik wurde vorab in [85] veröffentlicht.

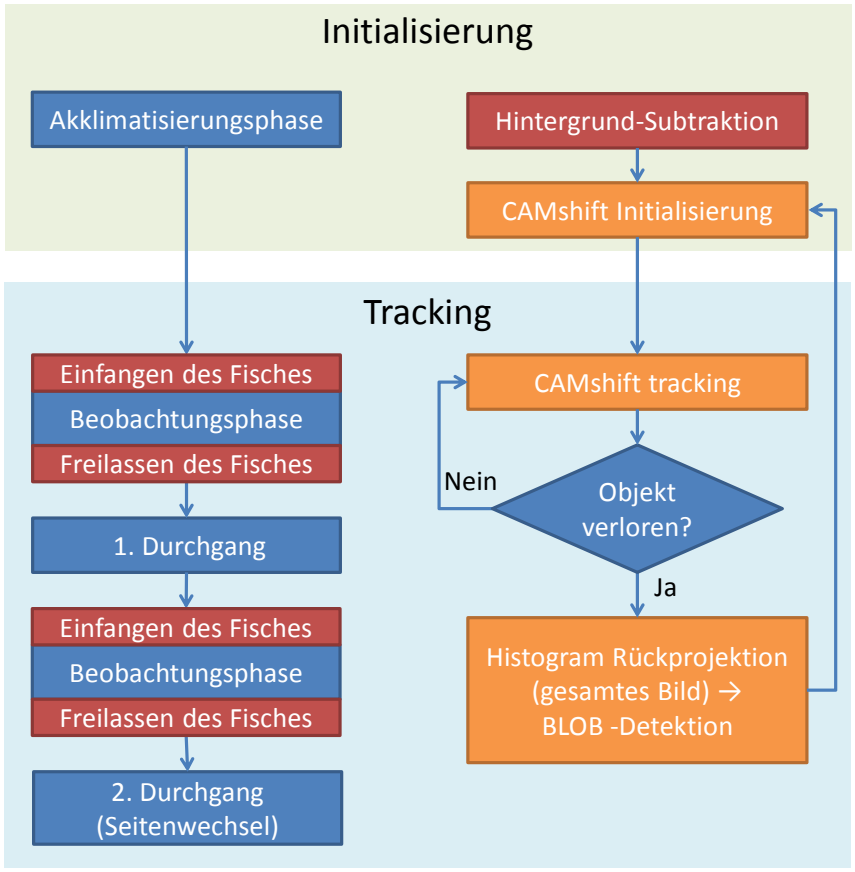


Abbildung 3.18: **Ablauf des Verfahrens während eines Partnerwahl-Versuches.** Nach der Initialisierung (grün hinterlegt) ist das Verfahren für den Versuch einsatzbereit (blau hinterlegt). Die rot eingefärbten Prozesse beinhalten einen manuellen Eingriff des Experimentators in das Aquarium.

3.8.2.2 3D-Positionsberechnung und Bewegungsschätzung

Die geschätzte 3D-Position P eines Fisches ermittelt aus den Pixelkoordinaten ${p'}_1$ und ${p'}_2$ der CAMshift-Tracker wird ebenfalls mit Hilfe des Ray-Tracing-Ansatzes aus Kapitel 3.5 berechnet. Dazu werden im ersten Schritt die Pixelkoordinaten mit Hilfe der Gleichung 3.42 in die Strahlen $\vec{x}_{p1}$ und $\vec{x}_{p2}$ umgerechnet. Die Umrechung beinhaltet auch die Lichtbrechung. Da es sich bei den Punkten ${p'}_1$ und ${p'}_2$ lediglich um die Mitte des Trackingfensters und nicht um einen definierten Punkt auf der Fischoberfläche handelt, ist es sehr unwahrscheinlich, dass sich die Strahlen treffen, sondern in einem geringen Abstand aneinander vorbeilaufen. Aus diesem Grund werden die Punkte auf beiden Strahlen gesucht, an denen sich diese am nächsten sind. Dazu wird das in [42, S.43ff.] beschriebene *Ray-to-Ray*-Verfahren genutzt. Der Mittelpunkt P zwischen dem Punkt C_1, welcher auf dem Strahl der Frontkamera liegt, und dem Punkt C_2 welcher, auf dem Strahl der Topkamera liegt, bildet gleichzeitig die 3D-Position des Fisches:

$$P = \frac{C_1 + C_2}{2} \tag{3.51}$$

Um auch im Fall einer kurzzeitigen Fisch-Verdeckung (keine Positionsdaten) eine geschätzte Position zu erhalten, wird die 3D-Position P zusätzlich mit Hilfe eines Kalman-Filters geschätzt. Eine Beschreibung des Filters ist in Kapitel 3.1.4.5 zu finden. Für den hier beschriebenen Einsatzzweck wurde ein sechsdimensionaler Zustandsvektor gewählt, der die Position und die Geschwindigkeit des Fisches beschreibt. Zur Berechnung des Übergangs von einem zum nächsten Berechnungsschritt wird ein Konstante-Geschwindigkeits-Modell gewählt. Bei diesem liegt die Annahme zu Grunde, dass sich das Objekt zwischen zwei Berechnungsschritten mit konstanter Geschwindigkeit bewegt. Für die Fischbewegung trifft dies nicht zu. Da das Filter jedoch mehrfach pro Sekunde aktualisiert wird, kann der resultierende Fehler vernachlässigt werden.

Daraus ergibt sich die Übergangsmatrix

$$F_k = \begin{pmatrix} 1 & 0 & 0 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 & dt & 0 \\ 0 & 0 & 1 & 0 & 0 & dt \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}. \tag{3.52}$$

Q_k und R_k wurden manuell gewählt. Die Elemente der Matrix Q_k wurden mit dem Wert $1 \cdot 10^{-5}$ und die Elemente der Matrix R_k mit dem Wert $5 \cdot 10^{-3}$ initialisiert. Die Zahlenwerte der Kovarianzmatrix des Messrauschens R_k ist dabei höher angesetzt worden als die Kovarianzmatrix des Systemrauschens Q_k, was zur leichten Glättung der Bewegung führt. Im Falle einer Verdeckung wird ausschließlich der Prädiktionsschritt durchgeführt. Nach erneuter Detektion des Fisches folgt der gewohnte Wechsel aus Korrektur und Prädiktion.

3.8.2.3 Fehlerdetektion und Reinitialisierung

Es kann vorkommen, dass einer oder beide CAMshift-Tracker ein falsches Objekt verfolgen, welches dieselbe Farbverteilung wie der initialisierte Fisch hat (z. B. ein sich spiegelnder Fisch an der Aquariumscheibe). Dies wird vonseiten des Trackers nicht erkannt, sondern muss durch einen zusätzlichen Mechanismus abgefangen werden. Dazu wurden zwei verschiedene Strategien entwickelt:

1. Wie in Unterkapitel 3.8.2.2 beschrieben, kreuzen sich die Mittelpunkt-Strahlen der CAMshift-Suchfenster $\vec{x}_{p1}$ und $\vec{x}_{p2}$ im optimalen Fall in der Mitte des Objektes oder verlaufen in diesem Punkt sehr dicht aneinander vorbei. Basierend auf dieser Annahme wird der Abstand der beiden Strahlen durchgängig mit einem manuell definierten Schwellwert (10 mm) verglichen. Sollte eine der beiden Tracker ein falsches Objekt verfolgen, erhöht sich der Abstand der Strahlen und ein Fehler wird detektiert.

2. CAMshift passt das Suchfenster durchgehend der detektierten Objektgröße an. Diese Eigenschaft wird zur Fehlerdetektion genutzt und minimale und maximale Schwellwerte für die Fenstergröße werden manuell

definiert. Ist einer der Werte der beiden Suchfenster (top und front Kamera) über- oder unterschritten, wird eine Fehlermeldung ausgelöst.

Nach einer Fehlerdetektion muss den Trackern erneut die korrekte Objektposition im Bild mitgeteilt werden. Dies gilt auch im Falle einer Verdeckung des Fisches (z. B. durch den Experimentator während der Platzierung des Plexiglas-Zylinders). Die Position kann zwar für einen kurzen Zeitraum geschätzt werden (siehe Kapitel 3.8.2.2), doch nach dem Zeitraum der Überbrückung muss die Objektposition erneut in den CAMshift-Trackern initialisiert werden. Zur Wiederauffindung des Objektes wird im ersten Schritt eine Histogramm-Rückprojektion auf das ganze Bild durchgeführt. Dies dauert länger als bei einem einzelnen CAMshift-Tracking-Schritt, bei welchem lediglich ein erweiterter Bereich um die letzte Position des Fisches berechnet wird. Dabei werden Bildbereiche, welche die im Histogramm vertretenen Farben beinhalten, hervorgehoben. Durch eine BLOB-Analyse werden die hervorgehobenen Bereiche segmentiert. Sollten pro Ansicht mehrere Segmente gefunden werden, werden die Abstände der Segment-Schwerpunkt-Pixelstrahlen aller möglichen Kombinationen aus beiden Ansichten geprüft. Die Mittelpunkte des Segment-Paares mit dem kürzesten Abstand werden genutzt, um die 2D-Koordinaten der CAMshift-Suchfenster zu aktualisieren.

3.9 Ergebnisse

Die Tests wurden in Aquarien mit den Abmessungen 600 mm x 300 mm x 300 mm und 1000 mm x 500 mm x 400 mm (Breite x Höhe x Tiefe) durchgeführt. Als Kameras standen zwei Gigabit Ethernet Farbkameras (Allied Vision Technologies, Prosilia GT1910c) mit einer Auflösung von 1920 x 1080 Pixeln zur Verfügung. Die Tests wurden mit männlichen und weiblichen Breitflossenkärpflingen (*Poecilia latipinna*) durchgeführt.

3.9.1 Kalibrierung und Kompensation der Lichtbrechung

Zu Beginn der Tests wurde das System mit Hilfe der Kalibrationssoftware kalibriert. Die Kalibrierung funktioniert vollautomatisch und kann auch von ungeübten Benutzern durchgeführt werden.

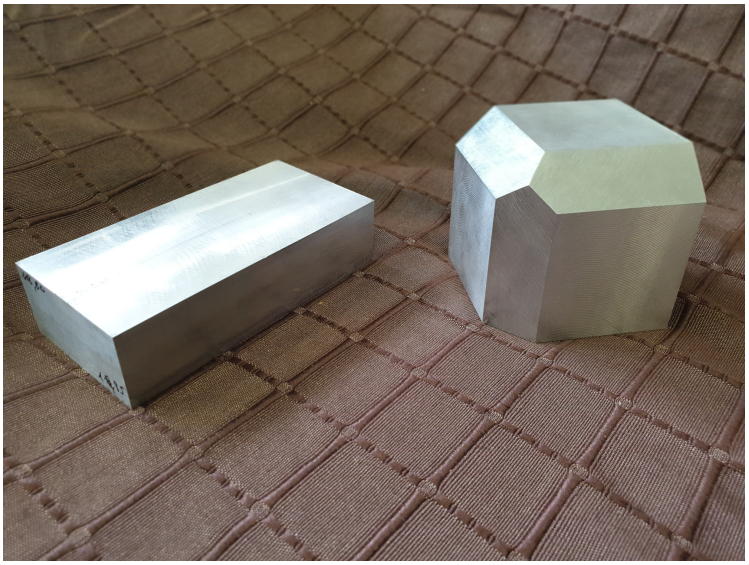


Abbildung 3.19: **Aluminiumblöcke zur Überprüfung der räumlichen Genauigkeit.** Die Blöcke haben eine Abmessung von 120 mm x 60 mm x 30 mm und von 70 mm x 70 mm x 70 mm.

Zur Bestimmung der räumlichen Genauigkeit des Systems wurden zwei Aluminium Blöcke (siehe Abbildung 3.19), welche mit einer Genauigkeit von 0,01 mm gefertigt wurden, mit dem System an unterschiedlichen Stellen des Aquariums vermessen. Zum einen wurden diese in den Ecken des Aquariums, deren Koordinaten bekannt sind, positioniert, um die absolute Positionsbestimmung zu überprüfen. Zum anderen wurden die Blöcke an weiteren, zufällig gewählten Positionen im Aquarium platziert, um anhand der gemessenen Länge die relative Genauigkeit zu testen. Die Blöcke wurde mit beiden Kameras in jeder neuen Position aufgenommen. Die Ecken der Quader wurden manuell in jedem Bild markiert und die 3D Position aus den referenzierten Eckpaaren nach dem in Kapitel 3.7 beschriebene Verfahren bestimmt.
Zur Bestimmung der relativen Genauigkeit wurden insgesamt 30 Längenmessungen durchgeführt. Dabei wurde je Quader die Länge der Kante gemessen, die von beiden Kameras eingesehen werden konnte. Die Länge ergab sich aus der Differenz des berechneten Start- und Endpunktes (in 3D-Koordinaten) der Kante. In den Tests wurde bei der Längenmessung ein durchschnittlicher Fehler von 0,31 mm bei einer Standardabweichung von 0,22 mm gemessen. Es wurden 20 absolute Positionsmessungen durchgeführt. Dabei wurde ein maxi-

maler Fehler von 2,1 mm gemessen. Zudem wurden diese Messungen auch bei ausgeschalteter Kompensation der Lichtbrechung durchgeführt. Dabei erhöhte sich der maximale Fehler der Positionsbestimmung auf 12,6 mm.

3.9.2 Detektion und Segmentierung

Die Detektion der Fische auf Basis des in Kapitel 3.6 beschriebenen Verfahrens wurde unter verschiedenen Belichtungsbedingungen, mit Fischen unterschiedlicher Größe und mit unterschiedlichem Bodengrund (Sand und Kies) getestet. Die Initialisierung des CB-Verfahrens mittels GMS-Verfahren war in allen Tests erfolgreich. Dabei korrelierte die Dauer der Initialisierung mit der Anzahl der Fische im Becken (je mehr Fische desto länger die Dauer), dauerte jedoch während der Tests nicht länger als 60 Sekunden. Die implementierte Schatten-Detektion arbeitete zuverlässig. Es stellte sich jedoch heraus, dass ein hoher Schwellwert der Schattendetektion (sehr dunkle Schatten) auch die Segmentierung negativ beeinflusst, da auch Teile des Fisches als Schatten erkannt wurden. Dies betraf insbesondere die fast transparenten Schwanzflossen der weiblichen Fische. In Abbildung 3.20 ist beispielhaft die Kontur des segmentierten Fisches gezeigt. Da der Hintergrund aus Sicht der Frontkamera homogener (Folie) als der Hintergrund aus Sicht der Topkamera war, war die Kontur der Frontansicht gleichmäßiger. Die Seitenflossen des Fisches in der Abbildung waren transparent und wurden deshalb nicht detektiert.

3.9.3 Tracking

Da beim Kontur-Tracking der *tracking-by-detection*-Ansatz genutzt wird, ist auch die Leistung dessen stark vom Ergebnis des Detektionsalgorithmus abhängig. Das hier beschriebene Verfahren, welches auf der Hintergrund-Subtraktion beruht, bildet eine solide Grundlage. Durch die schon während der Detektionsphase gebildeten Konturpaare (Top- und Frontansicht) konnte, selbst bei überlagerten Konturen in einer Ansicht, eine Verfolgung des Objektes durch Betrachtung der zweiten Kontur problemlos durchgeführt werden.

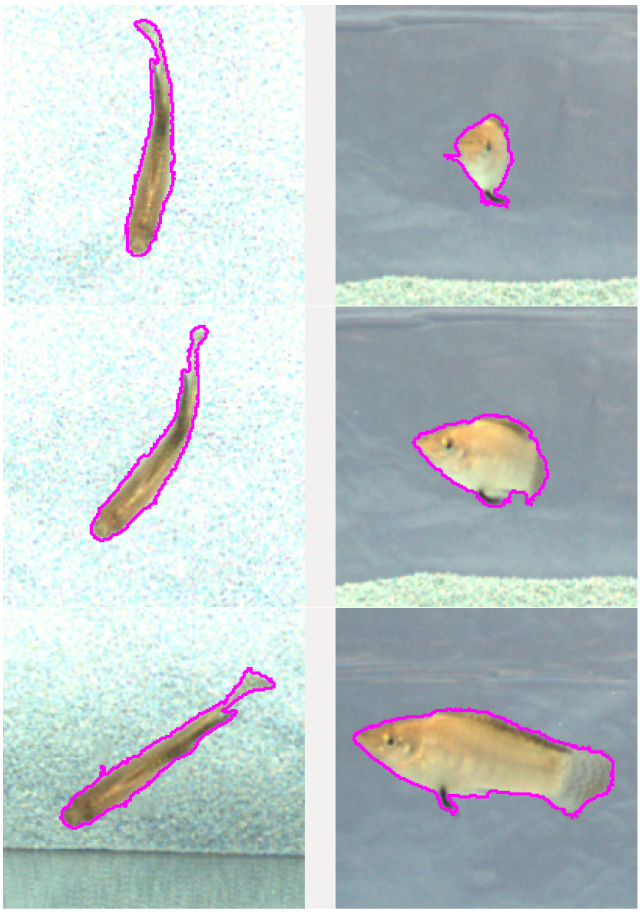


Abbildung 3.20: **Segmentierter Fisch.** Die Bilder zeigen beispielhaft die Segmentierung eines Fisches in unterschiedlichen Posen. Das linke Bild zeigt die Aufsicht und das rechte Bild die Frontansicht. Die lila Umrandung gibt die gefundenen Segmentgrenzen an.

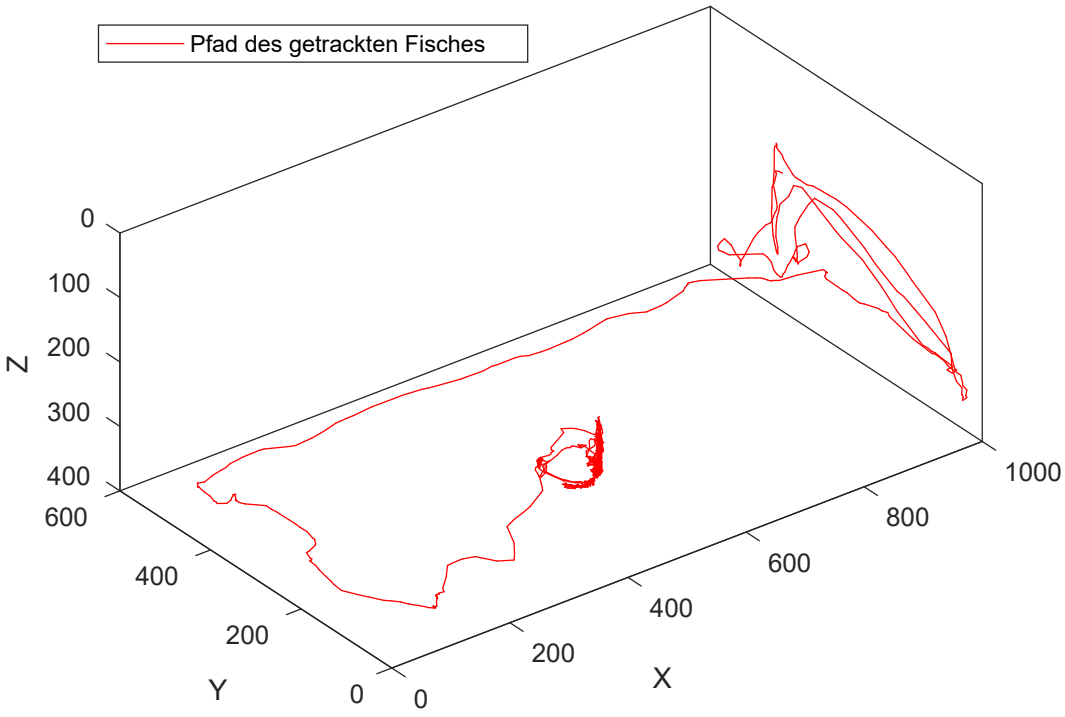


Abbildung 3.21: **Tracking-Pfad eines Fisches (in mm).** Das Diagramm zeigt einen 170 Sekunden Ausschnitt eines 3D-Fischpfades während eines Experimentes. Zu Beginn der Sequenz wurde der Fisch im Zylinder gefangen gehalten (115 Sekunden) und konnte sich somit nur im Zylinder selbst bewegen (roter Kreis in der Mitte). Nach der Freilassung schwamm er zum einen Ende des Aquariums und betrachtete den dort präsentierten Stimulus-Fisch und wechselte anschließend die Seite zum gegenüberliegenden Stimulus.

3.9.3.1 3D-Positionstracking

Das 3D-Positionstracking wurde in vielen Versuchsreihen mit interaktiver Steuerung (siehe Kapitel 4.4.3) getestet. Zu Beginn jedes Versuches wurde dabei der Versuchsfisch händisch und unter Verursachung von Wasserwellen eingefangen und freigelassen. Im Fall von Wasserwellen konnte der Fisch problemlos getrackt werden, da bei dem beschriebenen CAMshift-Verfahren lediglich die Farbverteilung berücksichtigt wird, welche sich durch die ungeordnete Brechung nicht wesentlich verändert. Auch bei Verdeckung des Fisches durch den Experimentator funktionierte die Positionsschätzung gut und der Fisch wurde erfolgreich wiedergefunden. Zur Bestimmung der 3D-Positionsgenauigkeit wurden zusätzliche Tests durchgeführt. Es wurden mehrere Bewegungssequenzen (2580 Frames, 20 Fps) eines weiblichen Breitflossenkärpflings aufgezeichnet. Dieser bewegte sich in einem 1000 mm x 500 mm x 400 mm Aquarium. Mit Hilfe des in Kapitel 3.7 beschriebenen Verfahrens zur Vereinigung der 2D-Konturen zum 3D-Modell wurden über diese Sequenzen hinweg 3D-Kastenmodelle erzeugt (siehe Abbildung 3.16), die manuell durch Abgleich mit der Videosequenz auf Vollständigkeit geprüft wurden. Da das 3D-Positionstracking keinen spezifischen Punkt auf dem Fischkörper verfolgt, sondern die höchste Pixeldichte im Bild der Histogramm-Rückprojektion, welche grob dem Fischmittelpunkt entspricht, wurde im Test die Position des Kastenmodell-Zentrums mit der berechneten Position verglichen. Bei den Testsequenzen lagen 99,5 % der berechneten Positionen innerhalb des Kastenmodells. Die mittlere Abweichung zwischen dem Zentrum des Kastenmodells und der berechneten Position belief sich auf 4,3 mm bei einer Standardabweichung von 1,9 mm. Ein Ausschnitt eines getrackten Pfades ist in Abbildung 3.21 zu sehen.

3.9.4 Berechnung der 3D-Repräsentation

Zur Visualisierung und Validierung der 3D-Repräsentation des Fisches wurde eine Software entwickelt, die sowohl das 3D-Kastenmodell (siehe Abbildung 3.23) als auch die Pixelstrahlen der Kontur in Echtzeit visualisieren kann (siehe Abbildung 3.22). Wie in Kapitel 3.7 beschrieben, werden die Konturpixel-Strahlen der beiden Kameras gebrochen und die Punkte entlang der Strahlen bestimmt, wo sich die beiden am nächsten sind. Der Mittelpunkt dieser wird

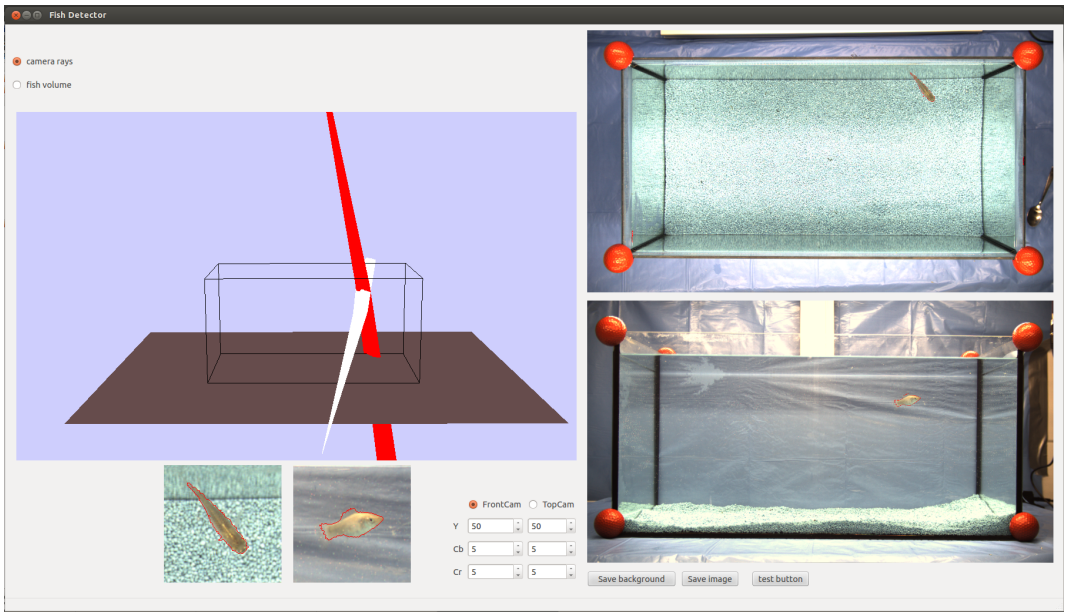


Abbildung 3.22: **Visualisierungssoftware.** Auf der linken Seite des Bildes ist ein virtuelles Modell des Aquariums zu sehen. In rot und weiß sind die Konturpixel-Strahlen der Top- und Frontkamera dargestellt. Diese Daten wurden aus den auf der rechten Seite gezeigten Bilder generiert. Sowohl die linke als auch die rechte Ansicht werden in Echtzeit aktualisiert. Diese Grafik wurde vorab in [87] veröffentlicht.

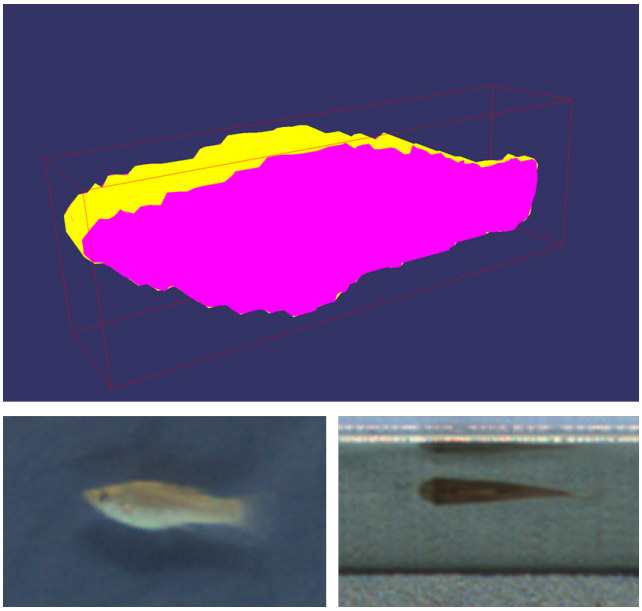


Abbildung 3.23: **3D-Kastenmodell eines Fisches.** Das obere Bild zeigt die Visualisierung des 3D-Kastenmodells eines Fisches. Dieser wurde aus den Fisch-Konturen der Front- und Topansicht (untere Bilder) generiert. Diese Grafik wurde vorab in [85] veröffentlicht.

als 3D-Koordinate genommen. Zur Berechnung einer Fehlermetrik, welche die Übereinstimmung der Strahlen aus Top- und Frontansicht beschreibt, wurden die Strahlen des Fischmauls (siehe Kapitel 3.6.4) verfolgt und deren minimaler Abstand berechnet. Zum Test wurde eine Videosequenz gewählt, in der ein Fisch mehrere Minuten durchs Becken schwamm. Die dabei gemessene Distanz zwischen den Strahlen betrug im Mittel 0,64 mm mit einer Standardabweichung von 0,72 mm.

3.9.5 Laufzeit

Die Laufzeit der vorgestellten Verfahren wurde auf Basis eines Rechners mit Intel i7-3770 CPU (4 Kerne, 3,4 GHz) und mit 8 Gb Arbeitsspeicher ermittelt. Die Hardware dieses Rechners stammt aus dem Jahr 2012 und kann heute (2018) mit der Leistung eines Rechners der unteren Mittelklasse verglichen werden. Die beiden synchronisierten Kameras lieferten Bilder mit einer Auflösung von 1920 x 1080 Pixel. Es wurde sowohl die Laufzeit des Konturtracking-Verfahrens mit anschließender 3D-Kastenmodell Generierung gemessen als auch

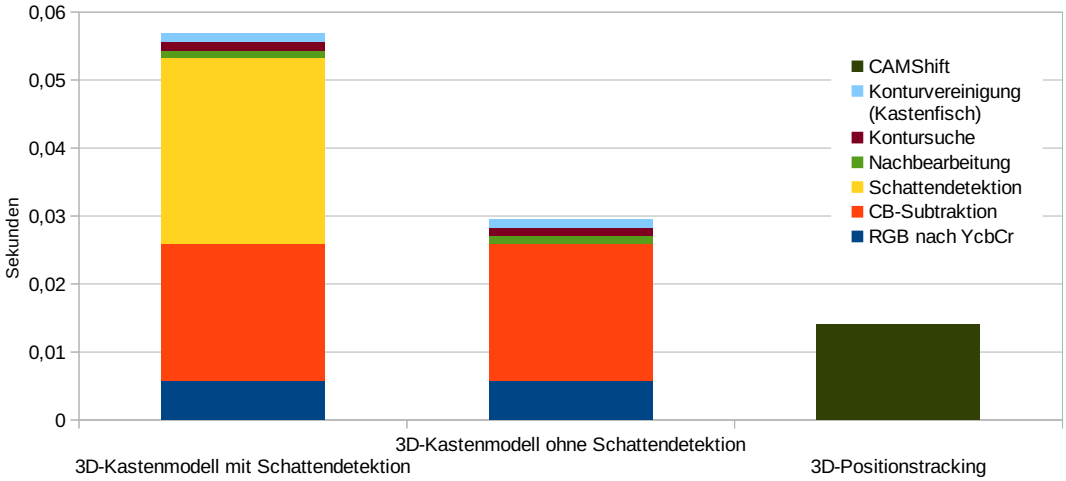


Abbildung 3.24: **Laufzeit der 3D-Kastenmodell Generierung und des 3D-Positionstrackings.** Die Laufzeiten sind nach den einzelnen Verfahrensschritten aufgeteilt.

das 3D-Positionstracking auf Basis des CAMshift-Verfahrens.
Das Verfahren zur Konturverfolgung wird global auf das Bild angewendet. Die Verfahrensschritt Farbraumwechsel, CB-Subtraktion, Schattendetektion, Nachbearbeitung und Kontursuche werden unabhängig von der Anzahl der Fische einmalig für jedes Bild durchgeführt. Lediglich die Laufzeit der Konturvereinigung steigt linear mit der Anzahl der Fische. Die gesamte Laufzeit wird sich somit auch mit zunehmender Anzahl von Fischen nicht stark erhöhen. Da die Schattendetektion eine zweite CB-Subtraktion erfordert, ist dieser Schritt sehr rechenaufwendig. Aus diesem Grund wurde zudem die Laufzeit ohne Schattendetektion ermittelt.
Anders als bei der Konturverfolgung, wird das CAMshift-Verfahren lokal auf die Umgebung eines einzelnen Fisches angewendet. Die Laufzeit des Verfahren wächst somit linear mit der Anzahl der Fische.
Die Messung ergab, dass die Erstellung des 3D-Kastenmodells ohne Schattendetektion 0,033 s dauert und somit eine Framerate von 30 Bildern pro Sekunde mit der beschriebenen Hardware möglich sind. Bei aktivierter Schattendetektion verringert sich dieser Wert auf 17,5 Bilder pro Sekunde. Das Verfahren zur Verfolgung der 3D-Position eines Fisches ist mit 71 Bilder pro Sekunde das schnellste Verfahren. Allerdings nimmt die Laufzeit bei mehreren Fischen zu.

3.10 Fazit

In diesem Kapitel wurden neue Verfahren zum 3D-Video-Tracking von Fischen und zur Erzeugung von einfachen 3D-Fisch-Approximationen unter Berücksichtigung der Lichtbrechung vorgestellt. Die Verfahren sind darüber hinaus echtzeitfähig und durch einen hohen Automatisierungsgrad einfach zu handhaben. Zudem wurden bei der Entwicklung die Standards der Partnerwahl-Experimente bei Breitflossenkärpflingen berücksichtigt.
Um eine hohe räumliche Genauigkeit zu erreichen, wurde erstmals (zum Zeitpunkt der Publikation [87], soweit dem Autor bekannt) ein Verfahren entwickelt, welches mit Hilfe eines Ray-Tracing-Ansatzes die Lichtbrechung des Aquariums beim Tracken von Fischen einbezieht. Zur Ermittlung der hierfür erforderlichen Kalibrierung und der Lichtbrechungsebenen wurde ein automatisches Kalibrierungsverfahren entwickelt, welches basierend auf einfachen, am Aquarium befestigten Markern arbeitet. Tests zeigten, dass eine relative Genauigkeit von +/- 0.3 mm und eine absolute Genauigkeit von 2,1 mm erreicht werden kann.
Durch die zweistufige Hintergrundsegmentierung ist es möglich trotz mehrerer Fische im Becken ein Hintergrundmodell zu generieren, welches keine Vordergrundobjekte (Fische) beinhaltet und somit eine pixelgenaue Segmentierung der Fische ermöglicht. Zudem wurde eine Schattendetektion als auch eine Methode zur Erkennung von Fischspiegelungen in den Seitenscheiben entwickelt. Dies rüstet das Verfahren für die speziellen Anforderungen des Trackens von Objekten innerhalb eines Aquariums.
Die Methode zur Vereinigung der 2D-Konturen zum 3D-Kastenmodells ermöglicht es in Echtzeit eine 3D-Visualisierung der aufgenommenen Fische zu erzeugen und dem Experimentator mit der entwickelten Visualisierungssoftware einen genauen Rundum-Einblick in das laufende Experiment zu geben.
Das zusätzlich entwickelte 3D-Positionstracking auf Basis des CAMshift-Algorithmus verfügt über eine automatische Initialisierung und kann auch bei welliger Wasseroberfläche und kurzzeitiger Verdeckung der Fische eingesetzt werden. Tests zeigten, dass die geschätzte 3D-Position des Fisch-Mittelpunktes 99,5% der Zeit innerhalb des Groundtruth-Kastenmodells lag und mit einer durchschnittlichen Abweichung von nur 4,3 mm Abweichung zum Zentrum

des Kastenmodells eine für die meisten Anwendungen ausreichende Präzision liefert.

Kapitel 4

Virtuelle, interaktive Fischstimuli zur Verhaltensforschung

Im Bereich der Verhaltensforschung bereitet die Standardisierung und Wiederholbarkeit von Experimenten Wissenschaftlern oft Probleme. Aus diesem Grund wurden in Teilbereichen der Forschung lebende gegen künstliche → Stimulus (Verhaltensforschung)-Tiere ausgetauscht. Neben der Möglichkeit Versuche zu standardisieren und deren Wiederholbarkeit zu verbessern, eröffnen künstliche, virtuelle Stimuli zudem die Möglichkeit auf einfache Weise die Morphologie und das Erscheinungsbild sowie das Verhalten zu definieren.
In diesem Kapitel wird eine neue Verfahrenskette zur Durchführung von visuellen Verhaltensexperimenten mit Fischen vorgestellt. Diese beinhaltet viele verschiedene und neuartige Funktionen, welche neue Möglichkeiten im Bereich der Verhaltensforschung eröffnen und die Standardisierung von Versuchen vorantreiben. Die vorgestellte Verfahrenskette beinhaltet alle notwendigen Methoden und Werkzeuge, um Fischstimuli zu designen, zu animieren und im Experiment mit lebenden Fischen einzusetzen. Die Software-Komponente zum Fisch-Design ermöglicht auf einfache und benutzerfreundliche Art Stimulus-Größe, -Färbung und -Morphologie manuell zu bestimmen und bietet zudem die Möglichkeit, Fische durch einen Zufallsalgorithmus und somit ohne menschliche Einflüsse gestalten zu lassen. Im Bereich der Stimulus-Animation bietet das entwickelte Verfahren eine neuartige Methode, um Stimuli mit Hilfe eines Gamecontrollers semiautomatisch zu animieren. Die auf diese Weise generierten Schwimmpfade sind unabhängig vom Stimulusmodell einsetzbar. Des

Weiteren wird eine Methode zur vollautomatischen Stimulus-Animation vorgestellt, die in der Lage ist die virtuellen Fische an beliebige Positionen im Aquarium zu steuern. Dies öffnet den Weg zur automatischen, interaktiven Stimulus-Animation, bei der aktiv auf die Außenwelt reagiert werden kann. Zur Präsentation der Animation umfasst die Verfahrenskette eine Software, welche Schwimmpfade und Stimulusmodelle kombiniert und die Abfolge verschiedener Animationen in Wiedergabelisten speichert. Jede Wiedergabeliste definiert die Animationsreihenfolge für Wiedergabegeräte während des Versuchs. Die einzelnen Komponenten und deren Zusammenspiel sind in Abbildung 4.1 zu sehen.
Teilthemen dieses Kapitels wurde vorab in [88] veröffentlicht. Das Gesamtsystem wurde in mehreren Versuchsreihen zur Partnerwahl mit lebenden Breitflossenkärpflingen validiert [48, 49].

4.1 Stand der Technik

Schon seit circa 80 Jahren werden künstliche, visuelle Stimuli in der Verhaltensforschung eingesetzt, um die Bedeutung von visuellen Informationen in der Tierkommunikation zu untersuchen. Die Vorteile von visuellen gegenüber lebenden Stimuli liegen besonders in der Möglichkeit Morphologie, Färbung und Bewegung frei zu gestalten und die Stimuli unabhängig vom individuellen Verhalten zu animieren. Des Weiteren können künstliche Stimuli zum Tierschutz beitragen, indem lebende Stimulus Tiere entsprechend der Leitprinzipien für Tierversuche (*The three R's - Reduce, Replace and Refine* [108]) partiell ersetzt ("**r**eplace") und damit deren Einsatz reduziert ("**r**educe") wird.
Bei den ersten Versuchen mit künstlichen Tierstimuli wurde auf einfache Modelle zurückgegriffen: Ter Pelkwijk und Tinbergen [138] verwendeten in der reizbiologischen Analyse einiger Verhaltensweisen von Dreistachligen Stichlingen (*Gasterosteus aculeatus*) Attrappen, die zum einen aus toten Exemplaren dieser Art und zum anderen aus Holz hergestellt wurden. Mit dem technischen Fortschritt veränderten sich die Methoden und Wissenschaftler erweiterten und verbesserten die Techniken. Baube, Rowland and Fowler [8] nutzten zum Beispiel einen elektrischen Motor, um einen künstlichen Dreistachligen Stichling mit konstanter Geschwindigkeit durch ein Testbecken zu bewegen, um die

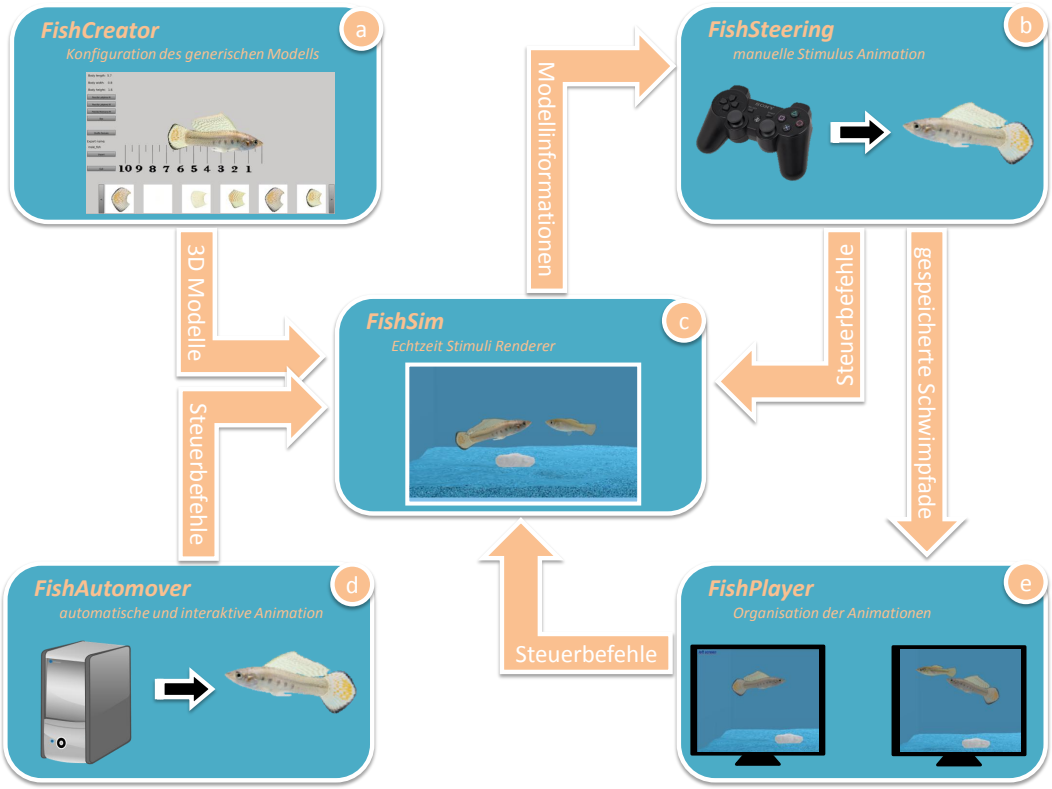


Abbildung 4.1: **Schematische Darstellung der Verfahrenskette.** (**a**) Konfiguration des generischen Modells. (**b**) Manuelle Animation des Stimulus mittels Gamecontroller. (**c**) Echtzeit Stimuli Renderer *FishSim*. (**d**) *FishAutoMover* dient der automatischen und interaktiven Steuerung der Stimuli. (**e**) Präsentation der Stimuli mittels *FishPlayer*. *FishPlayer* stellt für jeden Bildschirm eine eigene Playlist bereit.

natürliche Bewegung der Fische während der Experimente nachzuahmen. Daraus entwickelte sich der Einsatz von Roboterfischen: Faria et al.[43] benutzten einen Roboterfisch, der auf einem programmierten Pfad auf dem Beckenboden durch das Aquarium bewegt werden konnte. Butail et al. [18] gingen einen Schritt weiter und entwickelten einen ferngesteuerten, freischwimmenden Roboterfisch, der mit Hilfe eines visuellen Tracking-Verfahrens auf Kreisbahnen durch das Becken gesteuert wurde. Dabei konnte zum einen die Geschwindigkeit des Roboters und zum anderen die Frequenz des Schwanzflossenschlages verändert werden. Ziel der Arbeit war es, den Einfluss der Fortbewegung eines Zebrabärblings (*Danio rerio*) auf einen Schwarm derselben Art zu untersuchen. Ähnlich wie Faria et al. entwickelten Landgraf et al. [66] einen Fischstimulus, der computergesteuert in definierter Höhe über den Boden des Aquariums bewegt werden konnte. Im Gegensatz zu Faria et al. konnte der Fischstimulus bei Landgraf et al. zur Laufzeit beliebig durch das Becken gesteuert werden. Zusätzlich wurde sowohl der Roboter als auch der reale Guppy-Fischschwarm (*Poecilia reticulata*) im Becken durch ein Bildverarbeitungssystem getrackt und der Fischstimulus konnte auf dieser Basis aktiv auf das Verhalten des Schwarms reagieren. In einer späteren Arbeit wurde die Akzeptanz des Roboterfisches durch naturgetreue Augen und eine natürlichere Fortbewegung erhöht [65].

4.1.1 Bildschirmbasierte Stimuli

Im Gegensatz zu Roboterfisch-Stimuli zielt der Einsatz von bildschirmbasierten Stimuli auf die visuellen Reize während eines Experimentes ab. Taktile (z. B. Wasserströmungen und -wellen) und chemische Informationen werden von diesen nicht transportiert. In den ersten bildschirmbasierten Fischstimuli-Versuchen wurden Videoaufnahmen genutzt, die auf Röhrenbildschirmen, welche neben dem Testbecken platziert waren, gezeigt wurden. Auf diesem Wege war es möglich, jedem Testfisch dieselben Bewegungen und Verhaltensmuster eines Stimulus-Fisches zu zeigen und somit den Versuch unter gleichen Bedingungen zu wiederholen (vgl. [113] und [112]). Um auch Einfluss auf die Morphologie und das Erscheinungsbild des Stimulus nehmen zu können, setzten einige Wissenschaftler Viedeobearbeitungsprogramme ein, mit denen die Färbung (vgl. [74] und [75]) oder die Flossenlänge (vgl. [3]) verändert wurden.

Grundsätzlich bieten diese Programme jedoch nur eingeschränkte Möglichkeiten, um den Stimulus in einer Videosequenz zu variieren. 2D-Animationen bieten hingegen mehr Möglichkeiten die Merkmale eines Stimulus zu verändern. Bei Verfahren, die auf 2D-Animationen setzen, werden generell im ersten Schritt Bilder von echten Fischen mittels Bildbearbeitungssoftware bearbeitet. Typische Schritte während der Bildbearbeitung sind die Anpassung der Textur und Färbung, das Hinzufügen von transparenten Flossen oder das → Freistellen des Fisches vom Hintergrund des Bildes. Nachdem die Bildbearbeitung abgeschlossen ist, wird das Stimulusbild mit Präsentationssoftware (z. B. Microsoft PowerPoint [www.microsoft.de]) auf einfache Weise animiert (z. B. [7], [44] oder [140]). Zur Umsetzung dieser einfachen Technik benötigt man keine besonderen Kenntnisse im Bereich der Animation. Allerdings sind auch die Möglichkeiten dieser Verfahren beschränkt. Es lassen sich zwar translatorische Bewegungen in der Ebene durchführen; Drehungen des Stimulus im Raum oder Körperbewegungen wie Biegungen (z. B. beim Schwimmen) sind nicht möglich. Mehr Optionen bieten 3D-Animationen. Im Zuge des wachsenden Einflusses von 3D-Animationen im Film und durch die fortwährende Entwicklung von Computerspielen wurde professionelle und einfach zu benutzende 3D-Animationssoftware auch für viele Forschungsbereiche verfügbar.

4.1.1.1 Computeranimierte 3D-Stimuli

Auch die in diesem Kapitel vorgestellte Verfahrenskette hat das Ziel 3D-Fischstimulus Animationen zu erstellen und zu präsentieren. Im Folgenden wird ein Überblick über die bisherigen Arbeiten und Entwicklungen im diesem Bereich gegeben. Anschließend folgt eine technische Analyse der bisherigen Verfahren aufgegliedert in die Bereiche Stimulus-Design, -Animation und -Präsentation. 3D-Computer-Animationen werden schon seit mehr als 20 Jahren in der Fisch-Verhaltensforschung eingesetzt. McKinnon und McPhail [76] setzten in ihrer Arbeit im Jahr 1996 erstmals ein 3D-Computer-animiertes Dreistachliges Stichlings Männchen ein. Sie veränderten die Farbe der Kehle und des Kiemendeckels und analysierten das Aggressionsniveau des lebenden Testfisches hinsichtlich der Färbung des Stimulus. Bakker und Künzler [5] nutzten in ihrer Arbeit einen computeranimierten, männlichen Stichling, um die Partnerwahl-Präferenzen weiblicher Stichlinge zu testen. Sie konnten zeigen, dass ihre im

Versuch gewonnenen Ergebnisse mit denen der vorherigen Experimente mit lebenden und Video-Stimuli übereinstimmten. Modifizierte Versionen dieser Stichlingsanimation wurden auch erfolgreich in einigen Folgestudien eingesetzt [6, 64, 72, 73, 77].

Morris, Nicoletto und Hesselman [82] nutzten eine 3D-Animation eines männlichen Schwertträgers (*Xiphophorus continens*), um die Farbpräferenz von weiblichen Fischen bei der Partnerwahl zu untersuchen. Auch in [81] wurde diese Animation als Grundlage zur Erforschung des Einflusses der Körpergröße eines männlichen Fisches (virtueller Stimulus) bei der weiblichen Partnerwahl genutzt. Wong und Rosenthal [150] nutzten ebenfalls einen männlichen, virtuellen Schwerträger-Stimulus, um den Einfluss des Schwertes der Männchen bei der Partnerwahl zu untersuchen.

Zur Erzeugung einer Fischstimulus 3D-Animation ist technisches Verständnis und Erfahrung in diesem Bereich vonnöten. Aus diesem Grund wurde bei vielen der genannten Studien professionelle Unterstützung hinzugezogen. Dies verursacht teils hohe Kosten und ist nicht von jeder Forschungsgruppe zu stemmen. Um dem entgegenzuwirken veröffentlichten Veen et al. [142] eine kostenlose und benutzerfreundliche Software *anyFish*[1] zum Erstellen und Animieren von 3D-Fischstimuli. Ingley et al. [54] erweiterten diese Version (*anyFish2.0*) unter anderem, um die Möglichkeit 3D-Fischmodelle und Animationen mit anderen Forschungsgruppen zu tauschen (siehe auch [30]). Sie veröffentlichten neben der Software eine Anleitung und detaillierte Videotutorials. Culumber und Rosenthal [35] waren die Ersten, die *anyFish* nutzten, um den Einfluss des Polymorphismus der Schwanzflecken bei Papageienkärpflingen (*Xiphophorus variatus*) während der Partnerwahl zu untersuchen.

Butkowski et al. [22] erweiterten die Methode der virtuellen Stimuli und entwickelten eine Animation eines virtuellen, männlichen Schwertträgers *Xiphophorus birchmanni*, welche mit einem lebenden Weibchen interagieren konnte. Mit dieser Erweiterung konnten sie einer der größten Einschränkungen der bildschirmbasierten Stimuli aufheben: die fehlende Interaktion mit den lebenden Testfischen. Sie nutzten das Verfahren in einem Partnerwahlexperiment, in dem ein lebendes Weibchen zwischen einem virtuellen, artgleichen und einem virtuellen, artfremden Männchen entscheiden konnte. Während des Versuches

[1] http://swordtail.tamu.edu/anyfish

trackte eine Kamera das Weibchen in zwei Dimensionen (Ebene parallel zum Boden des Aquariums) und die virtuellen Männchen folgten diesem anhand der berechneten 2D-Position auf den Bildschirmen, die links und rechts neben dem Aquarium angeordnet waren.
Stowers et al. [132] gingen in ihrer Arbeit einen Schritt weiter und erzeugten eine →Virtual Reality-Umgebung für eine Zebrabärbling-Larve. Anders als in den vorhergehenden Versuchen wurde kein rechteckiges Aquarium genutzt, sondern eine mit Wasser gefüllte Halbkugel, auf die von unten durch mehrere Projektoren eine Animation projiziert wurde. Die Fischlarve wurde mittels vier Kameras, die oberhalb der Halbkugel angebracht waren, getrackt. Im Gegensatz zu Butkowski et al. [22] ist bei Stowers et al. die virtuelle Kamera, aus deren Sicht die Bilder der Animation gerendert (→rendern) werden, nicht statisch positioniert, sondern dynamisch an die Position des Fisches gekoppelt. Somit entsteht für die Fischlarve die Illusion einer virtuellen 3D-Umgebung. Diesen Aufbau nutzten sie, um die visuomotorischen Effekte auf die Fischlarven zu untersuchen. Dazu wurde eine virtuelle Umgebung mit Hindernissen auf die Halbkugel projiziert wurde. Zusätzlich nutzten sie den Aufbau um soziale Feedbackexperimente mit den Larven durchzuführen. Dazu wurde ein virtueller Stimulus (ebenfalls Zebrabärbling-Larve) in der virtuellen Welt dargestellt. Dieser virtuelle Stimulus war so programmiert, dass sein Schwimmverhalten zwischen zwei möglichen Zuständen durch einen Gewichtungsfaktor interpoliert werden konnte. Im ersten Zustand folgte der Stimulus seinem selbst bestimmten Pfad. Mit wachsendem Faktor wurde der Pfad des Fisches vom Pfad der echten Larve beeinflusst. Bei Erreichen des maximalen Faktorwertes folgte der virtuelle Fisch dem echten vollständig.
In diesem Kapitel wird die Entwicklung einer neuartigen Verfahrenskette zur Durchführung verhaltensbiologischer Versuche mit virtuellen, interaktiven Fischstimuli beschrieben. Die Verfahrenskette umfasst Programme zum Design eines Stimulus (*FishCreator*), zur manuellen (*FishSteering*) und automatischen (*Automover*) Animation und zur standardisierten Präsentation (*FishPlayer* mit der Visualisierung *FishSim*). Dabei wurden in allen genannten Bereichen neuartige Methoden entwickelt, die es auch unerfahrenen Anwendern ermöglichen, virtuelle Stimuli zu erzeugen, zu animieren und zu präsentieren. Darüber hinaus wurde mit dieser Software erstmals (soweit dem Autor bekannt)

ein System entwickelt, welches im dreidimensionalen Raum interaktive Partnerwahlversuche mit einem virtuellen Stimulus ermöglicht. Die Teilgebiete Design, Animation und Präsentation der Verfahrenskette werden in den folgenden Unterkapiteln genauer in den Stand der Technik eingeordnet.

4.1.1.2 Erzeugung des Stimulus

In den bisherigen Arbeiten wurden verschiedene Verfahren zur Stimulus-Erzeugung eingesetzt. McKinnon und McPhail [76] nahmen die Hilfe eines professionellen Animators in Anspruch. Zur Erstellung eines maßstabsgetreuen Modells nutzten sie morphologisch korrekte Maße des Fisches und erstellten aus Fotografien Texturen für den Stimulus. Auch Stowers et al. [132] beauftragten einen professionellen Animator zur Erstellung eines 3D-Zebrabärbling-Larvenmodells aus Fotografien. Bakker und Künzler [5] nutzten ein anderes Verfahren: Sie fixierten einen toten, männlichen Stichling mit Epoxidharz und schnitten diesen in 23 dünne Scheiben mit einer Dicke von je 1 mm. Jede dieser Scheiben wurde eingescannt. Die digitalen Bilder wurden entlang der Längsachse des Fischmodells in korrekter Reihenfolge angeordnet, um eine präzise 3D-Form des Fisches zu erhalten. Dieses 3D-Modell wurde mit Fotografien des Fisches texturiert. Die Flossen des toten Fisches wurden ebenfalls eingescannt und zur Texturierung genutzt.
Weit häufiger werden Größe, Form und Platzierung von Körperteilen (wie z. B. Flossen) aus Fotos rekonstruiert. Dazu wird der Fisch aus mehreren Ansichten fotografiert und diese 2D-Abbildungen anschließend zu einem 3D-Modell vereint. Dieses Verfahren wurde auch in [82] und [22] eingesetzt. Veen et al.[142], und Ingley et al. [54] entwickelten im Gegensatz dazu keinen einzelnen, in der Form nicht veränderbaren, virtuellen Fischstimulus, sondern erzeugten ein $\rightarrow$ Generisches Stimulusmodell, welches innerhalb der Software *anyFish* individuell anhand von morphologischen Daten angepasst werden kann. Die aktuelle Version von *anyFish* unterstützt eine begrenzte Anzahl von Fischarten. Dazu zählen der Stichling, Lebendgebärender Zahnkarpfen (*Poeciliidae*) und der Zebrabärbling. Um in der Software ein 3D-Modell eines Fisches zu erzeugen, wird eine seitliche Aufnahme des Fisches der Wahl benötigt, auf welchem 56 Landmarken manuell bestimmt werden müssen. Die Landmarken repräsentieren morphologische Merkmalspunkte des Fisches. *anyFish* erzeugt anschlie-

ßend aus diesen Informationen automatisch ein virtuelles Fischmodell.
Ähnlich wie bei Veen et al. [142] und Ingley et al. [54] bietet auch die in dieser Arbeit entwickelte Software *FishCreator* die Möglichkeit auf Basis eines generischen Stimulusmodells einen Stimulus zu gestalten (siehe Kapitel 4.2). Anders als bei den vorherigen Arbeiten ist es mit der Software jedoch möglich, Größe, Textur und Flossen intuitiv mittels Computermaus zu verändern. Des Weiteren bietet die Software als Alleinstellungsmerkmal einen Algorithmus zur Stimulus-Generierung per Zufall, um subjektive, menschliche Einflüsse auszuschließen.

4.1.1.3 Animation des Stimulus

Neben Morphologie und Aussehen spielt besonders ein natürlicher Bewegungsablauf eine wichtige Rolle, um die Akzeptanz des Stimulus zu erhöhen (vgl. [90]). In den bereits beschriebenen Studien wurden unterschiedliche Methoden angewendet: Bakker und Künzler [5] leiteten manuell die Bewegungen eines männlichen Stichlings bei der Balz mit Hilfe der → Rotoskopie von einer Videosequenz ab. Morris et al. [82] nutzten eine ähnliche Methode. Sie analysierten in 2700 Videoframes (→ Frame) die Bewegung eines männlichen Schwertträgers. Die Dauer spezieller Bewegungsmuster wurde bestimmt. Anschließend wurden diese mit Hilfe des virtuellen Stimulus mit gleicher Dauer nachgestellt.
Veen et al. und Ingley et al. [54, 142] implementierten verschiedene Animationsmöglichkeiten in der Software *anyFish*: Mit Hilfe der sogenannten → Key-Frame-Animation, wird der Stimulus manuell zu bestimmten Schlüsselposen (Key-Frames) bewegt. Diese Posen werden in definiertem Zeitabstand abgespeichert. Die Bewegungen zwischen den Key-Frames werden durch einen Algorithmus über die Zeit interpoliert. Somit kann man durch setzen einiger weniger Key-Frames eine längere Sequenz animieren. In *anyFish* können die Key-Frames zusätzlich mittels Rotoskopie aus einer Videosequenz abgeleitet werden. Als weitere Möglichkeit kann der Stimulus auf Basis von → Motion-Capture-Daten, welche durch eine Fremdsoftware erfasst wurden, animiert werden. Die so gewonnenen Bewegungspfade können editiert und mit anderen Forschern ausgetauscht werden. Da die in Butkowski et al. [22] benutzte Animation in Echtzeit und in Abhängigkeit von den Tracking-Informationen geren-

dert (→ rendern) werden muss, sind die zuvor beschriebenen Methoden nicht nutzbar. Stattdessen wurde eine Bibliothek mit 24 verschiedenen Schlüssel-Bewegungsanimationen (engl. key-movement animation) erstellt. Jede Sequenz stellt eine typische Bewegung des Fisches wie z. B. eine Vorwärtsbewegung oder eine Drehung dar. Diese können automatisch zu einer kompletten Bewegungssequenz, wie sie bei einem Versuch erforderlich ist, zusammengestellt werden. Das Animationssystem war mit einem 2D-Tracking-System verbunden, welches die Position eines Fisches in einem Aquarium verfolgte. Das Aquarium war zwischen zwei Bildschirmen, die jeweils eine Animation zeigten, positioniert. Während der Versuche animierte das System den virtuellen Stimulus so, dass dieser der Position des lebenden Fisches auf dem Bildschirm in Echtzeit folgte.

Stowers et al. [132] verknüpften das 3D-Modell der Fischlarve mit einem Animationsskelett bestehend aus 32 Knochen. Die Schwanzflossenbewegung wurde aus Tracking-Daten abgeleitet und mit Hilfe eines Animationstools auf das 3D-Modell übertragen. Die Bewegung der Larve wurde in zwei Phasen unterteilt: In der Abstoß-Phase wurde die virtuelle Larve durch einen Schwanzflossen-Schlag beschleunigt, in der nachfolgenden Phase schwebte der Stimulus und verlor an Geschwindigkeit. Diese Phasen wiederholten sich im 0,5-Sekunden-Takt.

Die Arbeit von Tu und Terzopoulos [141] ist nicht im Bereich der Verhaltensforschung angesiedelt, soll jedoch auf Grund ihrer Bedeutung für die Erzeugung von künstlichen, virtuellen Fischen hier aufgeführt werden. Sie entwickelten eine virtuelle, physikbasierte Unterwasserwelt, mit mehreren Fischen. Die Bewegung des Fisches wurde mit Hilfe eines Masse-Feder-Systems modelliert, welches durch externe Kräfte angeregt wurde. Die Forscher modellierten ebenfalls die Hydrodynamik, die während des Schwimmens auf die Fische wirkt. Des Weiteren wurden die virtuellen Fischen mit Verhaltensmustern ausgestattet: bei Futtereintrag in die virtuelle Welt, schwammen die virtuellen Fische zum Futter hin und wichen dabei reaktiv den unvorhersehbaren Bewegungen ihrer Artgenossen aus.

In der hier vorgestellten Verfahrenskette wird ein Softwaremodul zur manuellen Animation (*FishSteering*) und zur automatischen Animation (*Automover*) vorgestellt (siehe Kapitel 4.4). *FishSteering* bietet im Gegensatz zu den bisheri-

gen Methoden die Möglichkeit, den Stimulus intuitiv mittels Game-Controller zu steuern. Ein Algorithmus unterstützt den Benutzer, indem die Animation der Schwimmbewegung automatisch den Steuerbefehlen (z. B. *schwimme vorwärts*, *schwimme eine Kurve*, *schwimme runter/hoch*) angepasst wird und der Benutzer sich auf die Erstellung des Schwimmpfades konzentrieren kann. Das Modul zur automatischen Animation (*Automover*) bietet zum einen die Möglichkeit den Fisch zufällig durch das Becken zu steuern und das Balzverhalten eines männlichen Breitflossenkärpflings zu imitieren. Zum anderen umfasst die Methode auch die Möglichkeit, den Stimulus gezielt zu einer Position im Becken zu lenken. In Verbindung mit einem Tracking-System besteht die Möglichkeit, den virtuellen Fisch in Abhängigkeit von einem echten Fisch im angrenzenden Aquarium steuern zu lassen. Auch die Balzanimation lässt sich auf diese Weise in Abhängigkeit von der Position und Bewegung des echten Fisches in den Versuchsablauf eingliedern.

4.1.1.4 Präsentation des animierten Stimulus

Die Präsentation des Stimulus in einem Versuch ist der letzte Schritt in der Animationskette. In den meisten Arbeiten wurden die Animationen vor Beginn des eigentlichen Versuchs in eine Videodatei gerendert (→ rendern). Diese wiederum kann einfach auf Video-Wiedergabegeräten (vgl. [76]) oder mittels einer Video-Wiedergabe-Software auf einem Computer wie in [54, 142] wiedergegeben werden. In der Arbeit von Butkowski et al. [22] war es nicht möglich vorgerenderte Videos zu benutzten, da erst während der Laufzeit durch die Bewegung des lebenden Fisches entschieden wird, wo sich der Stimulus hinbewegen muss. Aus diesem Grund wurde die Animation in Echtzeit gerendert und auf dem Bildschirm präsentiert. Dazu setzten Sie eine → Spiel-Engine ein, die aus der Computerspieleentwicklung stammt. Auch bei Stowers et al. [132] muss die Animation systembedingt zur Laufzeit gerendert und präsentiert werden. Sie setzten in ihrer Software *Freemovr* das → Szenegraphen-System *OpenSceneGraph* ein. Ein ausführlicher Bericht über konzeptionelle und technische Aspekte des Stimulus-Designs, der Animation und der Präsentation ist in der Arbeit von Chouinard-Thuly et al. [30] zu finden.
Wie auch bei Butkowski et al. [22] und Stowers et al. [132] wird auch in der hier vorgestellten Methode die Animation in Echtzeit gerendert. Dies geschieht

zeitgleich auf zwei verschiedenen Monitoren mittels der Software *FishSim*. Der Schwimm- und Bewegungspfad kann dabei sowohl vor dem Versuch manuell aufgezeichnet als auch während der Ausführung automatisch durch *Automover* generiert werden. Als besonderes Merkmal ist dabei hervorzuheben, dass der Animationspfad unabhängig vom Modell animiert werden kann. Somit ist es möglich denselben Pfad mit unterschiedlichen Fischstimulus zu animieren und somit die Standardisierung der Versuche voranzutreiben. *FishPlayer* bietet darüber hinaus die Möglichkeit eine ganze Versuchsreihe zu planen. Dabei können sowohl die Animationspfade (Aufzeichnung, zufällige Bewegung oder Bewegung auf Basis von Tracking-Daten) als auch die Modelle und Aktionen (z. B. Balzen) für jede der beiden Bildschirmanimationen unabhängig konfiguriert werden.

4.2 Stimulus-Design

Die Erzeugung des Stimulus ist in dieser Arbeit in zwei Teile geteilt: Im ersten Teil werden zwei generische 3D-Modelle (männlich und weiblich) je Art erstellt. Diese repräsentieren zum einen die Morphologie der Art und zum anderen beinhalten sie das Animations-Skelett. Im zweiten Teil wird das generische Modell mittels benutzerfreundlicher Oberfläche konfiguriert. Im Folgenden wird die Erstellung eines generischen Stimulus-Modells anhand eines männlichen Breitflossenkärpflings gezeigt. Die verwendete Methode ist jedoch auch auf andere Fischarten anwendbar. Die *FishSim Animation Toolchain*[2] umfasst aktuell sieben generische Modelle, die auf diese Weise erzeugt wurden: *Gasterosteus aculeatus* (männlich), *Poecilia latipinna* (männlich und weiblich), *Poecilia mexicana* (männlich und weiblich) und *Poecilia reticulata* (männlich und weiblich).

4.2.1 Generisches 3D-Modell

Rosenthal [111] definiert in seiner Arbeit zwei unterschiedliche Methoden des 3D-Stimulus-Designs: beispielbasierte Animation (*example-based animation*) und parameterbasierte Animation (*parameter-based animation*). Das hier ent-

[2] `https://bitbucket.org/EZLS/fish_animation_toolchain`

wickelte generische Modell umfasst beide Ansätze. Eine Fotografie, die eine Längsseite des Fisches zeigt, wird genutzt, um die 2D-Form des Stimulus abzuleiten. Zur Erzeugung des 3D-Mesh (→ Mesh (Polygon)), werden Längenparameter wie Dicke, Länge und Höhe am lebenden Fisch gemessen und zur → Extrusion genutzt. Jede Flosse des Fisches (Rücken-, Brust- und Schwanzflosse) besteht aus einer Fläche, die dem Mesh zugeordnet ist und die Flossentextur abbildet. Die Fläche der Flossen ist großzügig gewählt, so dass diese später auch für großflächige Texturen ausreichend Platz bietet (siehe Abbildung 4.2), um so auch Stimuli außerhalb der artspezifischen Abmessungen zu erzeugen. Die Flächenbereiche, die nach Konfiguration des Stimulus nicht von einer Textur abgedeckt sind, werden vom → Renderer nicht berücksichtigt und sind in der späteren Animation nicht zu sehen. Die Flächen der Flossen werden anhand der seitlichen Fotografie des Fisches am Mesh positioniert. Die Augen des Stimulus werden als Halbkugel dem Mesh hinzugefügt. Die Halbkugeln haben eine hohe Transparenz und kommen so den echten Linsen des Fisches nahe.

Um den Stimulus im späteren Verlauf zu animieren, wird ein standardisiertes Skelett bestehend aus 36 Knochen (→ Rigging (Animation)) mit dem Mesh verbunden (siehe Abbildung 4.2). Die Anzahl der Knochen und deren Position wurde im Konsens zwischen Informatikern und Biologen an der Universität Siegen ermittelt und ist ein Kompromiss zwischen einer möglichst geringen Knochenanzahl und einer natürlichen Stimulusbewegung. Für die hier verwendeten männlichen Breitflossenkärpflinge werden zwei zusätzliche Knochengruppen hinzugefügt: Drei Knochen dienen dem Aufstellen der Rückenflosse und ein Knochen bewegt das Gonopodium des Fisches, welches das kopulatorische Organ der männlichen, lebendgebärenden Zahnkarpfen ist, zu denen auch die Breitflossenkärpflinge zählen. Grundsätzlich ist es möglich beliebige Knochen dem Skelett hinzuzufügen, um arttypische Bewegungen naturgetreu umzusetzen.

Jeder Knochen ist mit dem umliegenden Teil des Meshs verbunden. Mit Auslenkung des Knochens bewegt sich auch das verbundene Mesh. Die Verbindung zwischen Mesh und Knochen ist nicht statisch, vielmehr kann der Einfluss eines Knochens auf einen Knoten des Meshs beliebig angepasst werden. Ein Meshknoten, der zwischen zwei Knochen liegt, kann somit von beiden beein-

flusst werden, was einer natürlichen Verformung der Oberfläche nahekommt. In der hier verwendeten, frei verfügbaren Animationssoftware *Blender* (v. 1.71, `www.blender.org`) heißt dieser Prozess → weight painting und kann sowohl manuell als auch automatisch auf Basis der Entfernung zwischen Mesh und Knochen durchgeführt werden. Nach demselben Prinzip wurden für alle aktuell verfügbaren Arten die generischen Modelle erzeugt. Die Morphologie und die Texturen wurden artspezifisch angepasst, das verwendete Grundskelett wurde beibehalten. Die auf diesem Weg erzeugten generischen Modelle können mit Hilfe des *FishCreators* (siehe Kapitel 4.2.3) konfiguriert werden. Zu den hier beschriebenen Fischarten wurde zusätzlich eine 3D-Box generiert, die auf Grund ihrer Unbeweglichkeit über kein Animationsskelett verfügt. Die Box wurde in [49] genutzt, um die Bewegung und Form eines Stimulus zu entkoppeln und deren Einfluss auf Testfische zu testen. Die 3D-Modelle werden im DirectX Format (`.x`) exportiert. Dieses Format erwies sich als besonders geeignet, um 3D-Modelle verlustfrei aus Blender zu exportieren und in die verwendeten → Spiel-Engine *Irrlicht* (v. 1.81, `http://irrlicht.sourceforge.net/`) zu importieren.

4.2.2 Erzeugung der Fisch-Texturen

Zur Texturierung des Körpers und der Flossen des virtuellen Stimulus werden aus Fotografien ausgeschnittene → Texturen genutzt. Diese werden mittels → UV mapping auf das 3D-Modell projiziert. Die UV-Mapping Vorlagen, welche die 2D-Projektion der Flächen des 3D-Modells beinhalten, können vom generischen 3D-Modell mit dem Animationstool *Blender* extrahiert werden.
Die Fische in dem hier beschriebenen Beispiel wurden in einem kleinen Aquarium (40 cm x 40 cm x 12 cm, Wasserstand 10 cm) von der Seite mit gespreizten Flossen fotografiert (Canon EOS 600D, F/5.6, 1/250 s, ISO 200, Brennweite 55 mm). Als Beleuchtung dienten zwei Leuchtstoffröhren, die über dem Aquarium angebracht waren, und ein externer Kamerablitz (Nissin Speedlite). Um den Einfluss der Beleuchtung auf die Farbwerte der Fotografie zu reduzieren, wurde ein → Weißabgleich mittels Graukarte durchgeführt. Um die Wahrscheinlichkeit zu erhöhen, dass sich der Fisch mit seiner Längsseite zur Kamera ausrichtet, wurde eine weiße Kunststoffplatte in geringem Abstand parallel zur Frontscheibe des Aquariums aufgestellt.

Mit Hilfe einer Bildbearbeitungssoftware (GIMP [www.gimp.org]) wurden die Texturen ausgeschnitten und mit Bearbeitungsfunktionen wie Drehen, Skalieren und der in GIMP verfügbaren Käfig-Transformation auf die zuvor extrahierten UV-mapping-Vorlagen angepasst. Zudem wurde den Texturen der Flossen eine partielle Transparenz hinzugefügt, um diese naturgetreuer erscheinen zu lassen. Grundsätzlich können in diesem Schritt alle von den Bildbearbeitungsprogrammen bereitgestellten Funktionen zur Anpassung der Textur genutzt werden. Denkbare Anpassungen wären z. B. die Einfärbung der Textur oder das Hinzufügen von Mustern in Form von Punkten oder Streifen. Die so erzeugten Texturen werden als PNG-Datei (Portable Network Graphics) mit definiertem Namensformat abgespeichert. Im Gegensatz zu dem gängigen Bildformat JPEG bietet das PNG-Format zusätzlich die Möglichkeit Transparenz, wie sie für die Flossen erforderlich ist, abzuspeichern. Der Dateiname gibt dabei Aufschluss über Art und Position der Textur und die Textur wird anhand des Namens automatisch von der Konfigurationssoftware *FishCreator* dem entsprechenden Stimulus zugeordnet.

4.2.3 Konfiguration des generischen Modells und der Szene

Das zuvor erzeugte generische Stimulus-Modell kann mit Hilfe der Software *FishCreator* benutzerfreundlich über eine grafische Oberfläche konfiguriert werden. Das gewünschte generische Modell wird mit der arttypischen Standardgröße für Länge, Breite und Höhe geladen. Diese Maße können mit Hilfe der Software angepasst werden und die Einstellungen werden ad hoc in der Visualisierung übernommen. Des Weiteren können alle Texturen auf einfachem Wege per Maus getauscht werden. Dazu wird die zu verändernde Textur angeklickt und eine Bibliothek mit den möglichen Austausch-Texturen öffnet sich (siehe Abbildung 4.3). Nach Auswahl einer Textur wird diese automatisch auf das Fisch-Mesh gelegt. Es können die Texturen des Körpers, der Schwanzflosse, der Rückenflosse und der Brustflosse geändert werden. Zusätzlich verfügt die Konfigurationssoftware über eine Zufallsfunktion mit der alle Texturen willkürlich bestimmt werden. Damit ist es möglich einen Stimulus ohne menschlichen Einfluss zu designen. Der fertig gestaltete Stimulus wird abschließend gespei-

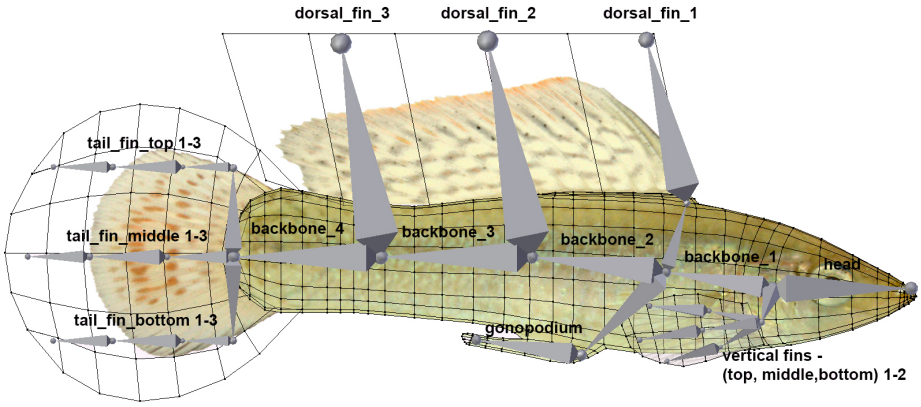


Abbildung 4.2: **Generisches Stimulus-Modell eines männlichen Breitflossenkärpflings.** Das 3D-Mesh repräsentiert die Morphologie des Fisches. Die grauen kegelförmigen Objekte stellen die Knochen des Modells dar, welche wiederum mit dem Mesh verbunden sind. Sowohl die Flossen als auch der Fischkörper sind mit einer Fotografie des Fisches texturiert. Diese Grafik wurde vorab in [85] veröffentlicht.

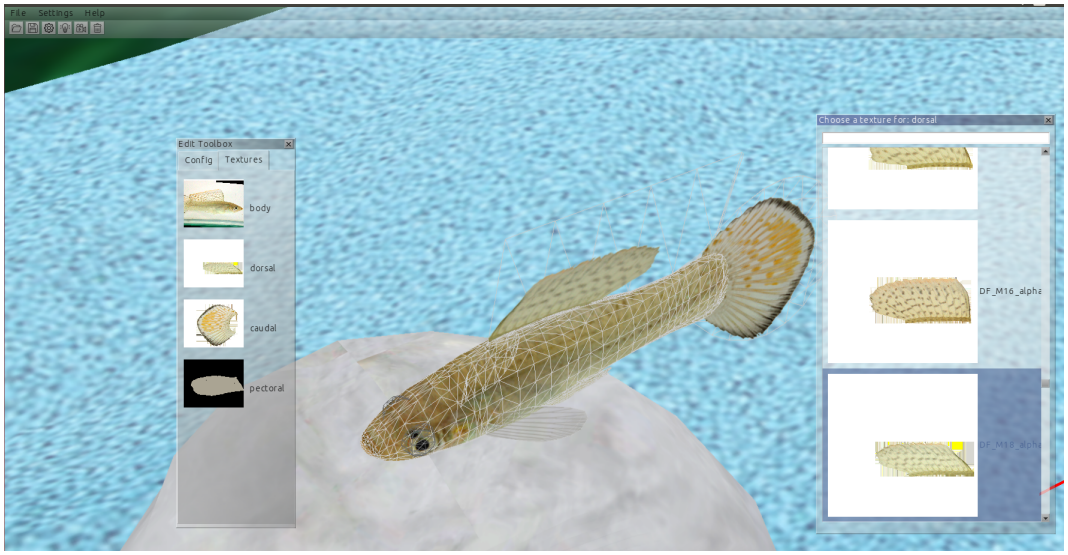


Abbildung 4.3: **Konfiguration des Stimulus.** Mit Hilfe der Software können die Texturen eines Stimulus-Modelles getauscht werden. Im Fenster auf der linken Seite kann der Teil des Stimulus ausgewählt werden, dessen Textur getauscht werden soll. Das Fenster auf der rechten Seite zeigt die verfügbaren Texturen an. Mit der Auswahl per Maus wird die Textur automatisch dem Stimulus-Modell hinzugefügt.

chert.

Neben dem Modell lässt sich in der Software auch die gesamte Szene gestalten. Dazu zählen →virtuelle Kamera (Position, Ausrichtung und Öffnungswinkel), Beleuchtung (Farbwert, Position und Ausrichtung), statische Objekte (Hinzufügen, Entfernen und Größenanpassung) und Stimuli (Position, Größe und Ausrichtung). Diese Änderungen werden, ähnlich wie bei der Konfiguration des Stimulus, in Echtzeit in der Visualisierung *FishSim* angezeigt. Nach Abschluss der Konfiguration wird die gesamte Szene in einem eigens für diesen Zweck erstellten Dateiformat (`.scene`) abgespeichert und kann sowohl in *FishSim* als auch mit Hilfe des *FishPlayer* (siehe Kapitel sec:FishPlayer) geladen werden.

4.3 Visualisierung - *FishSim*

FishSim ist die zentrale Komponente in der Verfahrenskette und basiert auf der →Spiel-Engine *Irrlicht*. Die Hauptaufgabe von *FishSim* ist die Visuali-

sierung der Stimuli. Dabei übernimmt *FishSim* auch Aufgaben wie die Umsetzung eines Kollisionsraums (Fische können in der Simulation nicht durch Wände schwimmen) oder die Berechnung und Darstellung des Fisch-Schattens. Die Software ist wie auch die anderen Komponenten der Verfahrenskette als → ROS-Node[3] implementiert. Alle zur Visualisierung notwendigen Informationen und Einstellungen können über die ROS-Infrastruktur bereitgestellt und vorgenommen werden. Genauere Informationen über die Infrastruktur des Systems sind in Kapitel 2.3.3 zu finden.
Stimulus-Modelle können direkt (als `.x`-Datei) oder indirekt zusammen mit einer Szene (`.szene`) geladen werden. Mit dem Modell werden auch das Animationsskelett und die beinhalteten Knochen geladen und intern gespeichert. Diese sind für andere Nodes der Verfahrenskette über einen → ROS-Service abrufbar. Die Hauptschnittstelle der Visualisierung bildet das → ROS-Topic `fish_action[_...]`, über welches die Animationsbefehle empfangen werden. Die Befehle kommen in Form der → ROS-Message `ActionArray`- bzw. `ActionArrayStamped`. Eine `ActionArray`-Message enthält wiederum mehrere `Action`-Messages. Jede`Action`-Message kapselt eine Bewegung eines Knochens, eine Stimulus-Position, eine Stimulus-Rotation oder eine Stimulus-Skalierung. Eine `ActionArray`-Message beinhaltet die Stellung aller Knochen und die Pose (Position, Ausrichtung und Skalierung) aller Stimuli in der Szene zu einem bestimmten Zeitpunkt (siehe Abbildung 4.4). Somit ist dieser Nachrichten-Typ nicht auf bestimmte Knochen oder Modelle beschränkt, sondern kann auch von nicht bekannten Stimulus-Modellen Bewegungsinformationen transportieren.

Wie alle ROS-Nodes kann auch *FishSim* auf einem verteilten System betrieben werden, welches über die ROS-Kommunikationsinfrastruktur aufgespannt ist. Die ROS-Kommunikation basiert auf dem nicht-deterministischen → TCP/IP-Protokoll. Somit ist die Ankunftszeit einer Message nicht definiert. Bei hoher Auslastung des Netzwerkes kann die Zeitspanne zwischen zwei Messages schwanken oder auch die Reihenfolge der Messages vertauscht werden. Dies würde die Animation negativ beeinflussen. Aus diesem Grund wurde die `ActionArrayStamped`-Message eingeführt, die zusätzlich einen Zeitstempel beinhaltet. *FishSim* prüft bei Eingang der Nachricht den Zeitstempel und hält in einem Puffer eine kleine Anzahl an Messages vor. Sollte eine Nachricht

[3]Eine genaue Beschreibung der Middleware ROS ist im Kapitel 2.3.2 zu finden.

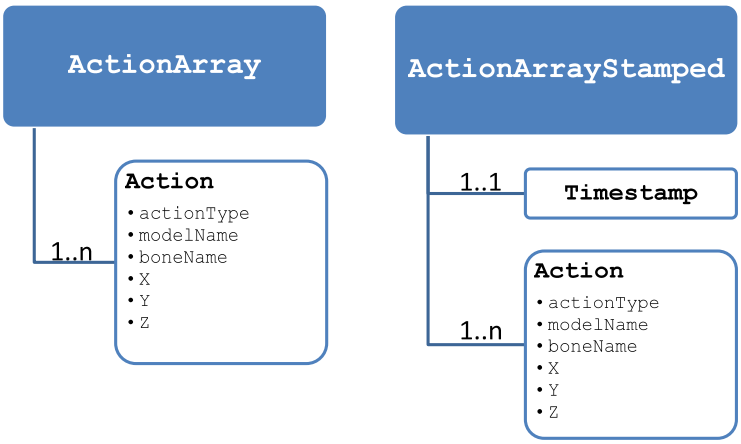


Abbildung 4.4: **Aufbau der `ActionArray`- und `ActionArrayStamped`-Message.** Sowohl `ActionArray` als auch `ActionArrayStamped` beinhalten n Messages vom Typ `Action`. `ActionArrayStamped` beinhalten zudem einen Zeitstempel.

nicht rechtzeitig ankommen oder fehlen, interpoliert *FishSim* zwischen den benachbarten Nachrichten, so dass die Animation ohne Abriss flüssig läuft.
In der aktuellen FishSim-Version wird die virtuelle Szene 60-mal pro Sekunde gerendert (→ rendern) und somit die Animation 60-mal pro Sekunde aktualisiert. Diese Aktualisierungsrate ist für die meisten Fischarten ausreichend (vgl. [46, 95]). Grundsätzlich ist es jedoch auch auf einfachem Wege möglich die Bildwiederholungsrate für Tiere mit schnellerem Sinnesorgan oder für spezielle Anzeigegeräte anzupassen.

4.4 Stimulus Animation

Im Rahmen der hier präsentierten Arbeit wurden drei verschiedene Verfahren der Stimulus Animation implementiert. Neben der halbautomatischen Animation, bei welcher der Stimulus mit Hilfe eines Gamecontrollers gesteuert wird, wurde ein automatisches Verfahren entwickelt, welches den Stimulus zufällig durch das Aquarium steuern oder gezielt zu einer Position schwimmen lässt. Dabei sind auch zusätzliche Bewegungen wie das Aufstellen der Rückenflosse oder das Bewegen des Gonopodiums durch die automatische Animation abgedeckt. Auf dieses System setzt die interaktive Animation auf. Die zu-

vor zufällig gewählten Zielpositionen werden nun durch die interaktive Steuerung gewählt und der interaktive Fischstimulus kann auf Bewegungen eines getrackten Fisches reagieren. Die hier vorgestellte Methode umfasst neben der automatischen Annäherung und Verfolgung des echten Fisches auch eine Balz-Animation.

4.4.1 Halbautomatische Animation

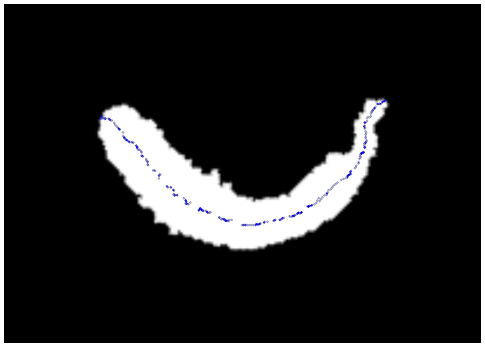
Zur halbautomatischen Animation des Stimulus wurde die Software *FishSteering* entwickelt. Diese ist ebenfalls als ROS-Node umgesetzt und ermöglicht es dem Benutzer auf einfache Weise Stimulus-Animationen mit Hilfe eines *Sony PlayStation* Controllers (DualShock 3, `www.playstation.com`) zu erstellen. Dabei bietet die Software sowohl die Möglichkeit den Stimulus in Echtzeit zu steuern und zu präsentieren als auch die Animation für eine spätere Präsentation zu speichern. *FishSteering* ist eng mit *FishSim* verknüpft: Die erzeugten Stimulus-Befehle werden als `ActionArray`-Message verpackt und während der gesamten Animation in *FishSim* angezeigt. Die zur Animation benötigten Stimulus-Informationen (z. B. Anzahl, Name oder Knocheninformationen) werden automatisch mittels eines ROS-Service aus *FishSim* geladen.
Vor Beginn der eigentlichen Animation können die Stimuli mittels Game-Controller in der virtuellen Umgebung platziert werden. Im Falle von mehreren Stimuli werden diese sukzessive animiert: Nach Erstellung eines Animationspfades mit dem ersten Stimulus, wird die bisherige Animation zurück auf den Startpunkt gesetzt und die Animation des zweiten Stimulus kann beginnen. Währenddessen läuft die zuvor aufgenommene Animation des ersten Stimulus parallel mit und der Animator hat einen Überblick über alle Stimuli und deren Schwimmpfade. Somit ist es möglich auch Bewegungen zwischen zwei oder mehreren Stimuli aufeinander abzustimmen und so Verhalten wie Folgen oder Balzen umzusetzen. Durch dieses Bedienkonzept können mit *FishSteering* nahezu beliebig viele Stimuli animiert werden.
Um sich während der Animation ganz auf die Steuerung des Stimulus konzentrieren zu können, wird die Animation der Fisch-Schwimmbewegung (Körperbiegung und Flossenschlag) durch einen Algorithmus übernommen, welcher genauer in Unterkapitel 4.4.1.1 beschrieben ist.
Grundsätzlich bietet *FishSteering* drei verschiedene Schwimm-Modi an: Vor-

wärtsschwimmen, kurvenförmiges Vorwärtsschwimmen und auf der Stelle drehen. Alle diese Modi können stufenlos mit den Joysticks des Game-Controllers gesteuert werden. Die Schwimmgeschwindigkeit bewegt sich dabei für die in diesem Beispiel verwendeten Breitflossenkärpflinge zwischen 0 und 40 cm/s. Die Drehgeschwindigkeit für das Kurvenschwimmen sowie für das Drehen auf der Stelle bewegen sich in einem Bereich von 0 bis 200 °/s und die Drehgeschwindigkeit um die Nickachse des Fisches (Hoch und Runterschwimmen) bei 0 bis 30 °/s. Um eine stetige Drehung des Fisches zu gewährleisten, werden die Drehgeschwindigkeiten um die Gier- und Nickachse des Stimulus mittels → Proportionalregler geregelt. Die hier genannten Werte wurden durch eine Videoanalyse echter Breitflossenkärpflinge ermittelt und auf Basis persönlicher Erfahrung bei der Steuerung optimiert. Grundsätzlich können diese Werte je nach Art und Versuchsaufbau beliebig angepasst werden.
Nach Abschluss der Schwimmpfaderstellung hat der Benutzer zudem die Option einzelne Körperteile zu steuern. Im Falle des Breitflossenkärpflings können die Rückenflosse aufgestellt, das Gonopodium geschwenkt oder die Brustflossen bewegt werden. Dies geschieht ebenfalls in einem weiteren Durchgang der Animation.

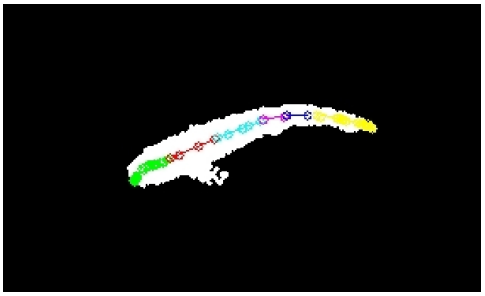
4.4.1.1 Automatische Animation der Schwimmbewegung

Die Animation einer authentischen Schwimmbewegung kann je nach Fischart sehr aufwendig sein; dies ist mit dem hier vorgestellten Konzept der einfachen Bedienung nicht vereinbar. Aus diesem Grund wurde eine von Smielik, dem Autor und Kuhnert [126] vorgestellte Methode eingesetzt, die basierend auf einem mathematischen Modell die Körperbiegung und Schwanzflossenbewegung des 3D-Stimulusmodells entsprechend der Geschwindigkeit, Beschleunigung und Richtung animiert. Im Folgenden wird diese Methode vorgestellt.

Zur Entwicklung der mathematischen Funktion wird mittels der in Kapitel 3.6 beschriebenen Methode die Silhouette eines Fisches aus der Topansicht über viele Frames einer Schwimmsequenz extrahiert. Entlang der Silhouetten-Längsachse wird eine Mittellinie geschätzt (siehe Abbildung 4.5, eine genaue Beschreibung ist [126] zu entnehmen). Zur Verknüpfung der Silhouettenbewegung und der Bewegung des virtuellen Stimulus, werden die Knochen des virtuellen Stimulus (siehe Abbildung 4.2) entsprechend ihrer Länge den Teilbe-



(a) Mittellinie durch die Fischsilhouette



(b) Verteilung der Animationsknochen entlang der Mittellinie der Fischsilhouette. Jede Farbe repräsentiert einen anderen Knochen des Animationsskelettes.

Abbildung 4.5: **Rekonstruktion der Mittellinie und Knochenverteilung.** Beide Grafiken stammen aus [126].

reichen der Mittellinie zugeteilt (Abbildung 4.5). Da Breitflossenkärpflinge zum Schwimmen hauptsächlich eine Schwanzflossenauslenkung auf einer Ebene (zur linken und rechten Seite) vollziehen, ist auch die hier beschriebene Funktion auf die 2D-Auslenkung des Körpers und der Schwanzflosse beschränkt. Diese wird entlang der Längsachse durch eine Polynomfunktion ($\rightarrow$ Polynom) $P(x)$ (Gleichung 4.1) approximiert. Die Polynomfunktion wird durch Lösung eines Gleichungssystems (siehe Gleichung 4.2) bestimmt. Die Anzahl der Punkte auf der Fischmittellinie C entsprechen der Anzahl der Gleichungen des Systems. x beschreibt dabei die Position entlang der Mittellinie, i den Grad der Funktion, c die Polynomkoeffizienten und y die Auslenkung nach links und rechts.

$$P(x) = \sum_{i=0}^{p} c_i x^i \tag{4.1}$$

$$\begin{cases} c_0 + c_1 x_1 + c_2 x_1^2 + \cdots + c_p x_1^p = y_1 \\ c_0 + c_1 x_2 + c_2 x_2^2 + \cdots + c_p x_2^p = y_2 \\ \vdots \\ c_0 + c_1 x_C + c_2 x_C^2 + \cdots + c_p x_C^p = y_C \end{cases} \tag{4.2}$$

Da eine einzelne Polynomfunktion nicht jede Mittellinie der Schwimmsequenz gut repräsentiert, werden zwei verschiedene Polynomfunktionen zur Abbildung genutzt: die erste Polynomfunktion $P_1(x)$ wird an die Fischmittellinie während des Ausschlags von der Nullposition (kein Ausschlag der Flosse, keine Körperbiegung) bis zum maximalen Ausschlag angenähert. Die andere Polynomfunktion $P_2(x)$ approximiert die Fischmittellinie während der Bewegung vom maximalen Ausschlag hin zur Nullposition. Um nun einen stetigen Übergang zwischen den beiden Polynomfunktionen zu gewährleisten und einen Sprung in der Animation zu vermeiden, wird über die Zeit eine → Spline-Interpolation ($S(x,t)$) durchgeführt, die einen stetigen Übergang zwischen den beiden Polynomfunktionen $P_1(x)$ und $P_2(x)$ schafft.
Der periodische Teil der Schwimmbewegung wird durch eine Sinusfunktion approximiert, die von der Zeit t und dem Faktor B, der sich auf die Periode auswirkt, abhängig ist.

$$y(x,t) = S(x,t)sin(Bt) \tag{4.3}$$

Zur Bestimmung der Auslenkung aller Knochen werden die Knochen-Endpunkte in Gleichung 4.3 eingesetzt und anschließend der Winkel zwischen den Knochen berechnet.
Beim Schwimmen einer Kurvenbahn wird entsprechend der Drehrichtung des Stimulus ein Winkel auf jeden Animationsknochen aufaddiert. Somit wird auch auf der Kurvenbahn eine kontinuierliche Schwimmbewegung gewährleistet.

4.4.2 Automatische Animation

Neben der halbautomatischen Animation bietet die entwickelte Software die Möglichkeit der automatischen Animation. Die automatische Animation setzt auf die halbautomatische Animation auf: Die zuvor vom Benutzer durch das Gamepad gegebenen Steuerbefehle werden nun von einer Software gesetzt, der ROS-Node *FishAutomover*. Wie auch bei der halbautomatischen Animation wird bei der automatischen Animation die Flossenbewegung und die Körperbiegung durch die in Kapitel 4.4.1.1 beschriebene Methode generiert.

Das Grundprinzip der automatischen Animation basiert auf einer automati-

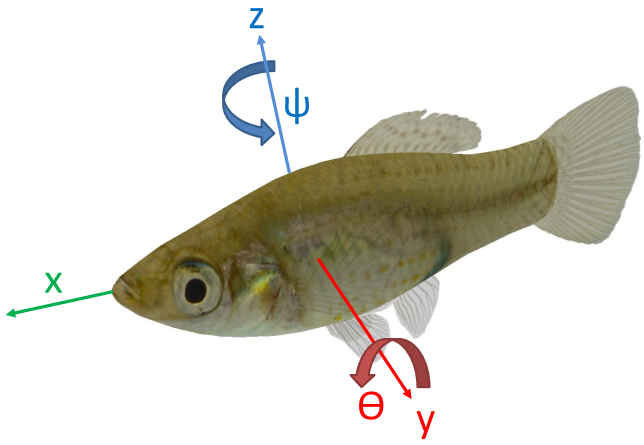


Abbildung 4.6: **Rotationsachsen des Fischstimulus.** Der Winkel θ beschreibt die Drehung um die Nickachse und der Winkel ψ um die Gierachse des Stimulus.

schen Annäherung des Stimulus an eine Pose (Position und Ausrichtung). Die Annäherung ist dabei in drei Teilbereiche unterteilt: Annäherung des Gierwinkels (ψ), Annäherung des Nickwinkels (θ) und Annäherung der Position $(x, y, z)^T$. Die beiden Drehwinkel werden mit Hilfe zweier $\rightarrow$Proportionalregler angenähert. Dazu wird im ersten Schritt die Abweichung zum Zielwinkel für jede Achse separat bestimmt. Der so berechnete Fehler wird im folgenden Schritt mit einem Proportionalitätsfaktor (V_p) multipliziert und zur aktuellen Reglerausgangsgröße hinzuaddiert (siehe Gleichung 4.6). V_p kann je nach Fischart angepasst werden. Eine Erhöhung des Proportionalitätsfaktors führt zur beschleunigten Reaktion des Reglers und damit zur schnelleren Drehung des Stimulus.
Die Reglung der Schwimmgeschwindigkeit ist sowohl von der Drehung des Fisches als auch von der Entfernung zum Zielpunkt abhängig und unterliegt folgender Annahmen:

- Die Vorwärtskraft des Stimulus ist bei hoher Entfernung zum Zielwegpunkt hoch und nahe des Zielwegpunktes niedrig.
- Während der Drehung des Stimulus ist die Geschwindigkeit niedrig, bei geradlinigem Schwimmpfad hoch.

Um beide Einflussgrößen in die Geschwindigkeitsregelung einzubeziehen,

werden zwei Faktoren gebildet, wovon der Faktor für die Entfernung zum Wegpunkt ($f_e(t)$) mit einem Drittel und der für die Drehung ($f_d(t)$) mit zwei Dritteln zum Zeitpunkt t in die Berechnung eingeht. Sowohl $f_e(t)$ als auch $f_d(t)$ liegen im Wertebereich $[0, 1]$ und berechnen sich in Abhängigkeit zu den zuvor manuell festgelegten Werten für die maximale Entfernung im Becken d_{max} und die maximale Reglerausgangsgröße $d_{\psi_{max}}$. d gibt den Abstand zwischen aktueller Position und Zielwegpunkt an. $d(t)_\psi$ gibt die aktuelle Reglerausgangsgröße des Proportionalreglers aus.

$$f_e(t) = \frac{d(t)}{d_{max}} \tag{4.4}$$

$$f_d(t) = \frac{d(t)_\psi}{d_{\psi_{max}}} \tag{4.5}$$

Zudem wird die Geschwindigkeit zusätzlich durch einen Zufallsparameter beeinflusst, um eine natürliche Schwankung zu erreichen. Die Grundgeschwindigkeit (v_b) wird bei Eingang eines jeden neuen Wegpunktes zufällig zwischen zwei zuvor manuell festgelegten Schwellwerten (untere Schwelle - niedrige Schwimmgeschwindigkeit, obere Schwelle - hohe Schwimmgeschwindigkeit) bestimmt. Die Geschwindigkeit ($v(t)$) berechnet sich wie folgt:

$$v(t) = \frac{2v_b}{3} f_d(t) + \frac{v_b}{3} f_e(t). \tag{4.6}$$

Um unnatürlich schnelle Geschwindigkeitsänderungen zu vermeiden, wird der Geschwindigkeitswert vor Applikation mit Hilfe eines → Gauß-Filters geglättet. Im Modus der automatischen Animation werden zufällig gewählte Wegpunkte mit der beschriebenen Methode angenähert. Sobald ein Wegpunkt erreicht ist, wird ein neuer zufällig bestimmt.

4.4.3 Interaktive Animation

Anders als bei der automatischen Animation ist die Stimulusbewegung bei der interaktiven Animation von der Bewegung des echten Fisches abhängig. Die dazu nötigen 3D-Positionsinformationen des realen Fisches wird durch das in Kapitel 3.8.2 vorgestellte 3D-Trackingsystem bereitgestellt. Das reale Aquarium wird von zwei Kameras observiert, die oberhalb und vor dem Becken angebracht sind. Auf der linken und rechten Seite des Aquariums befindet sich

jeweils ein Anzeigengerät, auf dem die Animationen präsentiert werden (siehe Abbildung 2.2, S. 14). Die Abmaße der Objekte in der virtuellen Umgebung werden durch manuelle Messung auf dem Anzeigengerät auf die reale Größe geeicht, sodass im Folgendem alle Berechnungen zwischen virtueller und realer Szene in einem Koordinatensystem durchgeführt werden können.
Die hier vorgestellte interaktive Animation ist speziell auf der Balz des männlichen Breitflossenkärpflings abgestimmt, bietet jedoch viele Methoden, die für andere Anwendungsfälle benutzt werden können.
Die interaktive Animation startet, sobald ein echtes Weibchen in die nähere Umgebung des virtuellen Stimulus' (nahe des Anzeigengerätes) kommt. Der virtuelle Stimulus nähert sich diesem in der virtuellen Welt an und erscheint auf dem Anzeigengerät größer. Anschließend folgt der Stimulus dem Weibchen in der vertikalen Ebene (Anzeigenebene). Bleibt der weibliche Fisch stehen, beginnt der virtuelle Fisch mit dem Balztanz. Dazu stellt er seine Rückenflosse auf und schwimmt vor dem Weibchen auf und ab.
Die interaktive Animation basiert auf der automatischen Animation und erweitert diese zum einen um Bewegungsmuster, die speziell auf die Interaktion mit lebenden Fischen abgestimmt sind, und zum anderen um eine interaktive Balz-Animation. Die verschiedenen Zustände der Animation wurden mit Hilfe eines → Zustandsautomats umgesetzt. Die Zustände und deren Übergänge werden in Abbildung 4.7 gezeigt. Im Folgenden werden die Zustände genauer erläutert:

AUTO Dieser Zustand umfasst die automatische Animation aus Kapitel 4.4.2. Der virtuelle Stimulus bewegt sich zufällig durch das Becken.

SWIM_CLOSER Der echte Fisch befindet sich in einer Zone nahe des Anzeigengerätes (Reizzone). Der virtuelle Stimulus nähert sich schnell der Position des echten Fisches. Dazu werden die Koordinaten des Trackingsystems in die Koordinaten der virtuellen Animation umgerechnet.

SLIDE Der echte Fisch bewegt sich vor dem Anzeigengerät und der virtuelle Stimulus versucht diesem mit schneller Bewegung zu folgen.

TURN_LEFT Diese Bewegung ist Teil der Balz. Der virtuelle Stimulus zeigt dem echten Fisch seine linke Seite und die Rückenflosse.

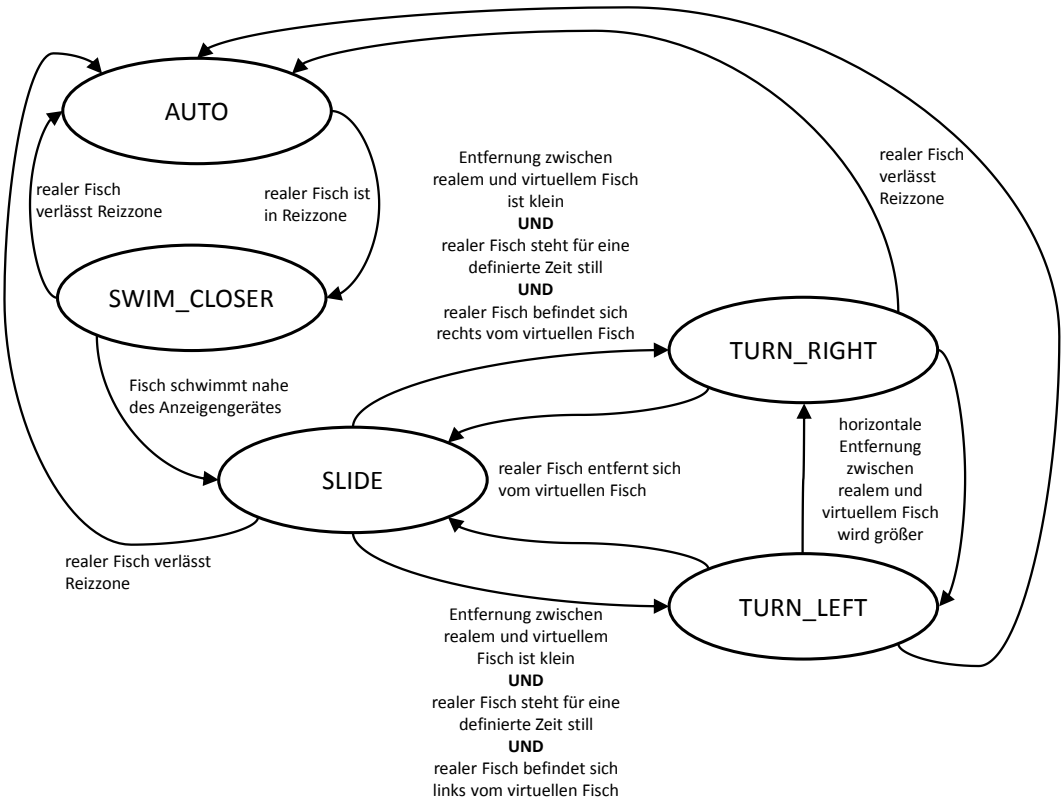


Abbildung 4.7: **Ablaufdiagramm der interaktiven Steuerung.** Das Diagramm stellt die verschiedenen Zustände und deren Zusammenspiel während der interaktiven Animation dar.

TURN_RIGHT Diese Bewegung ist Teil der Balz. Der virtuelle Stimulus zeigt dem echten Fisch seine rechte Seite und die Rückenflosse.

In den Zuständen **SWIM_CLOSER** und **SLIDE** nähert sich der Stimulus ähnlich wie im Zustand **AUTO** einem Wegpunkt an. Dieser Wegpunkt ist jedoch nicht statisch, sondern beschreibt die getrackte Position des echten Fisches und bewegt sich somit kontinuierlich. Würde man in diesem Zustand die Annäherung aus Kapitel 4.4.2 nutzen, würde sich der Fisch unnatürlich verhalten und sich stets in Richtung des nächsten Wegpunktes (Position des Fisches) ausrichten. Zudem ist es in der Phase der Annäherung wichtig, dass der Stimulus den schnellen Bewegungen des echten Fisches folgen kann. Aus diesem Grund wird in den Zuständen **SWIM_CLOSER** und **SLIDE** die Sensitivität der Regler erhöht, was zu einer schnellen Reaktion und einer zeitnahen Verfolgung des realen Fisches führt. Bleibt der lebende Fisch stehen, sind die Regler jedoch zu empfindlich. Dies führt nach Erreichen des Wegpunktes dazu, dass die Regler durchgehend versuchen die Lage des Stimulus zu korrigieren, damit dieser horizontal und vertikal optimal zum Betrachter ausgerichtet ist. Diese Bewegung wirkt unnatürlich und weicht vom Verhalten des echten Fisches ab. Um diesem zu begegnen, wird die Wirkung mit zunehmender Nähe zum realen Fisch verringert. Hierzu wird die horizontale (d_h) und vertikale Entfernung (d_v) zum echten Fisch im Verhältnis zum manuell definierten, maximalen Entfernungswert gesetzt und zur Skalierung der Regelgröße genutzt. Die Geschwindigkeitsregelung entspricht der in Gleichung 4.6 beschriebenen Berechnung. Lediglich die Grundgeschwindigkeit v_b wird mit einem höheren Wert festgelegt und nicht mehr zufällig gewählt.
Um während der interaktiven Animation in das Geschehen eingreifen zu können, sind in der Software einige ROS-Services umgesetzt. Zu diesen zählen Services zu der Aktivierung und Deaktivierung der automatischen Rückenflossenanimation, der Gonopodiumanimation oder dem Wechsel zurück in den **AUTO**-Zustand.

4.5 Stimulus Präsentation

Der reibungslose Ablauf der Stimulus-Präsentation spielt eine wichtige Rolle bei Partnerwahl-Experimenten. Dabei ist darauf zu achten, dass alle zu-

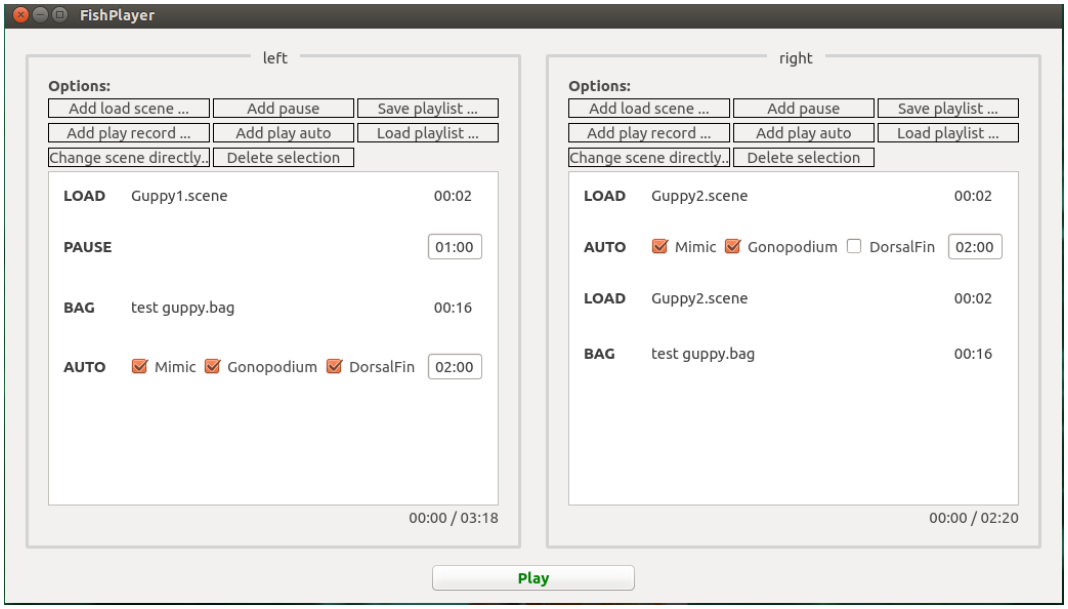


Abbildung 4.8: **FishPlayer.** Jedes der zwei Fenster zeigt eine Playlist für ein Anzeigengerät. Die Playlist-Einträge können über die Schaltflächen im oberen Bereich hinzugefügt werden. Für den Playlist-Eintrag *AUTO* kann durch Setzen der Hacken die Interaktion (*Mimic*) und die automatische Animation des Gonopodiums (*Gonopodium*) und der Rückenflosse (*DorsalFin*) aktiviert werden.

vor manuell erstellten Animationen bzw. die automatischen oder interaktiven Animationen in richtiger Abfolge auf dem richtigen Anzeigengerät präsentiert werden. Im Falle der hier beschriebenen Partnerwahl-Experimente werden zwei Anzeigengeräte verwendet, auf denen je nach Experiment eine Sequenz von verschiedenen Animationen gezeigt wird. Die Inhalte der Animationen variieren je nach Experiment. Oft werden jedoch pro Durchlauf verschiedene Stimuli abwechselnd auf beiden Anzeigengeräten gezeigt.

Um den Experimentator während des Experimentes zu entlasten und den geregelten Ablauf zu gewährleisten, wurde die Software *FishPlayer* erstellt. Wie auch die anderen Komponenten der Verfahrenskette ist FishPlayer als ROS-Node umgesetzt und ist modular einsetzbar. Mit dem Start von *FishPlayer* werden automatisch zwei Instanzen von *FishSim* und *FishAutomover* gestartet. Dabei bietet der *FishPlayer* die Möglichkeit Größe und Position der Animationsfenster zu konfigurieren. Die Software selbst stellt für jedes Anzeigengerät eine eigene Playlist bereit (siehe Abbildung 4.8), auf der sowohl

unterschiedliche Typen von Animationen als auch Steuerbefehle für die jeweilige *FishSim*-Simulationsinstanz hinzugefügt werden können. Die möglichen Optionen werden im Folgenden aufgezählt und erläutert:

Add play record Wie aus dem Playlist-Konzept von Unterhaltungssoftware bekannt, können mit dieser Option vorhandene Animationen (siehe Kapitel 4.4.1) der Playlist in chronologischer Reihenfolge hinzugefügt werden. Die Animationen müssen als → ROS-Bag vorliegen und können mit *FishSteering* erzeugt werden. Die Datei enthält die Animationsbefehle in Form von *ActionArray[Stamped]*-Messages. Während der Wiedergabe sendet *FishPlayer* die Nachrichten zur jeweiligen Anzeigen-Instanz über die ROS-Message-Infrastruktur. Somit ist auch eine Wiedergabe auf anderen im Netzwerk befindlichen Geräten möglich. Die Animationssequenz ist unabhängig vom Stimulusmodell. Dieses wird vor Beginn des Animationseintrages mit der Szene geladen. Dies ermöglicht es eine zuvor abgespeicherte Animation mit beliebigen Stimuli zu präsentieren und somit eine hohe Standardisierung der Experimente zu erreichen.

Add play auto Diese Funktion fügt eine automatische oder interaktive Animationssequenz ein, die zur Laufzeit durch *FishAutomover* erzeugt wird. Die erzeugten Animationsbefehle werden in *ActionArray[Stamped]*-Nachrichten verpackt und über die ROS-Infrastruktur an *FishSim* gesendet. Der Experimentator hat darüber hinaus die Möglichkeit festzulegen, ob die Animation auch die automatische Bewegung der Rückenflosse oder des Gonopodiums umfassen soll. Auch die interaktive Animation kann mit Hilfe dieses Eintrages gesetzt werden (Auswahl **Mimic** in Abbildung 4.8). Zudem kann für jeden Eintrag dieser Kategorie die Animationslänge bestimmt werden.

Add load scene Diese Option dient dem Laden einer zuvor konfigurierten Szene und der beinhalteten Stimuli (siehe Kapitel 4.2.3). Dieser Eintrag wird je nach Experiment vor Beginn einer neuen Animation gesetzt, um die Stimuli auszuwechseln oder die virtuelle Umgebung auszutauschen.

Change scene directly Neben dem Szenenwechsel zwischen zwei Animationen in der Playlist kann die Szene auch direkt gewechselt werden.

Add pause Zusätzlich kann zwischen zwei Animationen eine Pause eingefügt werden. Diese bietet sich an, wenn ein virtuelles Aquarium für eine bestimmte Zeit leer stehen soll.

Vor Beginn eines Experimentes definiert der Experimentator eine Playlist für jede Anzeige. Die Playlists sind dabei meist synchronisiert, um einen zeitgleichen Wechsel der Animation auf beiden Displays zu gewährleisten. Durch die genaue Ablaufplanung der Animationen mittels *FishPlayer* vor Beginn des Experimentes wird der Experimentator während des Versuchs entlastet und kann sich auf den Umgang mit den Versuchstieren konzentrieren.

4.6 Ergebnisse

Gierzewski et al. [49] nutzten die hier vorgestellte Verfahrenskette im halbautomatischen und automatischen Animationsmodus in mehreren Experimenten. Während der Validierung mit lebenden Breitflossenkärpflingen konnte die Akzeptanz der erzeugten 3D-Fischanimationen nachgewiesen und die Durchführbarkeit von Versuchen mit der entwickelten Verfahrenskette bestätigt werden. Es konnte gezeigt werden, dass weibliche Testfische Animationen eines 3D-Männchens einem leeren Aquarium gegenüber bevorzugten. Zudem stellte sich heraus, dass Weibchen die gleiche Zeit vor einem männlichen 3D-Stimulus, vor einem lebenden Männchen oder vor einer Videowiedergabe eines lebenden Männchens verbrachten, wenn diese zusammen mit einem leeren Aquarium (jeweils als Animation, real oder in einer Videosequenz) in einem Auswahlexperiment präsentiert wurden. Dies belegt, dass Weibchen bevorzugt Zeit mit dem gezeigten Männchen verbringen, unabhängig davon welche Methode gewählt wird. Dieses Ergebnis unterstreicht, dass die vorgestellte Animation genauso attraktiv wie lebende Fische oder Videos von lebenden Fischen ist.
In weiteren Experimenten wurden den Testfischen 3D-Männchen und virtuelle 3D-Boxen sowohl stillstehend als auch in Bewegung präsentiert (siehe Abbildung 4.9). Dabei konnte gezeigt werden, dass lebende Weibchen die bewegte Animation bevorzugten, selbst wenn es sich um eine schwimmende Box handelte. Dies bestätigt die Verwendbarkeit und Akzeptanz der vorgestellten Animationsmethode mittels Gamecontroller. Zusätzlich zeigten die Experimente, dass schwimmende, virtuelle Fischstimuli einem schwimmenden Box-Stimulus

Abbildung 4.9: **3D Modell eines Fisches und einer Box.** Die Box kann ähnlich wie der Fisch durch das virtuelle Aquarium schwimmen.

vorgezogen werden. Dies weist darauf hin, dass die weiblichen Fische zwischen den verschiedenen virtuellen Stimuli unterscheiden können. Darüber hinaus deuteten die Tests darauf hin, dass männliche Testfische zwischen weiblichen und männlichen 3D-Fischen unterscheiden können.
Diese Ergebnisse bestätigen, dass die Methode zur Erstellung von virtuellen 3D-Stimuli ermöglicht realistische, virtuelle Breitflossenkärpfling-Modelle zu erzeugen, die von lebenden Fischen akzeptiert werden und zudem ausreichende Informationen über geschlechtsspezifische Eigenschaften transportieren können. Die Validierung von Gierszewski et al. [49] bestätigt somit, dass die vorgestellte Verfahrenskette ein leistungsfähiges Werkzeug für Partnerwahl-Experimente mit Breitflossenkärpflingen ist.
In einer weiteren Versuchsreihe wurde der Einfluss eines *Gravid Spots*, einem dunklen Fleck, der bei trächtigen, weiblichen Breitflossenkärpflingen nahe der Afterflosse auftritt, im Experiment zum →Kopieren der Partnerwahl untersucht [48]. Dazu wurde der Textur eines virtuellen, weiblichen Stimulus mit Hilfe einer Bildbearbeitungssoftware ein dunkler Fleck im Bereich der Afterflosse hinzugefügt. Wie bei Versuchen zum Kopieren der Partnerwahl üblich, präsentierte man dem weiblichen Testfisch einen einzelnen, männlichen Stimulus und ein Stimulus-Fischpaar. Der weibliche Stimulus wurde in je einem Testdurchgang mit und ohne *Gravid Spot* gezeigt. In dem Experiment zeigte sich, dass der *Gravid Spot* zu keiner signifikanten Änderung beim Kopieren der Partnerwahl führte. Im Gegensatz zu vorherigen *Gravid Spot*-Experimenten

mit lebenden Fischen, bei denen der Fleck mit Tattoo-Tinte unter die Haut injiziert wurde (vgl. [10]), bietet dieses Verfahren eine nicht-invasive Alternative zur invasiven Manipulation.
Die interaktive Animation wurde in mehreren Experimenten von Ahmad et al. getestet [2]. In den Partnerwahl-Experimenten wurden sowohl weiblichen als auch männlichen Breitflossenkärpflingen virtuelle Stimuli des jeweils anderen Geschlechts präsentiert. Die eine Hälfte der Stimuli wurde durch die Methode zur automatischen Animation (siehe Kapitel 4.4.2) gesteuert. Die männlichen Stimuli zeigten dabei in zufälligen Zeitabständen Balzverhalten (Abbildung 4.10). Die andere Hälfte der Stimuli reagierte interaktiv auf den Testfisch. Sobald sich dieser dem Anzeigengerät näherte, wurde der Stimulus durch die automatische Animation zum Testfisch gesteuert und folgte diesem (siehe Abbildung 4.11). Die männlichen Stimuli führten zudem einen Balztanz auf, sobald der Testfisch an einer Position verharrte. Zur Auswertung der Experimente wurde die Zeit gemessen, welche die Testfische in zuvor definierten Zonen vor dem Anzeigengerät verbrachten. Es konnte gezeigt werden, dass weibliche Testfische sich sowohl für die interaktiven als auch für die nicht-interaktiven männlichen Stimuli interessierten und keine Angst vor den interaktiven Stimuli zeigten. Für Breitflossenkärpflinge konnte in den Versuchen kein signifikanter Unterschied zwischen den interaktiven und nicht-interaktiven Stimuli festgestellt werden. Im Nachgang zu den in [2] durchgeführten Experimenten wurden vier zufällig ausgewählte Versuchsaufzeichnungen der Experimente von Ahmad et al. (zwei Durchläufe mit interaktiver und zwei mit nicht-interaktiver Animation) in Hinblick auf die durchschnittliche Entfernung zwischen dem lebenden Fisch und dem virtuellen Stimulus in der 15 cm-Zone vor dem Bildschirm bestimmt. Das virtuelle und das reale Aquarium teilten sich ein Koordinatensystem und die Entfernungen konnten durch einfach Subtraktion der Positionswerte (Stimulus und lebender Fisch) berechnet werden. Dabei ergab sich, dass der durchschnittliche Abstand zwischen interaktivem und lebendem Fisch 3,6 cm betrug, wohingegen die durchschnittliche Entfernung zwischen nicht-interaktivem Stimulus und lebendem Fisch 22 cm betrug. Dies deutet darauf hin, dass die interaktive Steuerung sehr effektiv den virtuellen Stimulus an den realen Fisch annähert und somit zumindest die Funktionalität des Systems belegt.

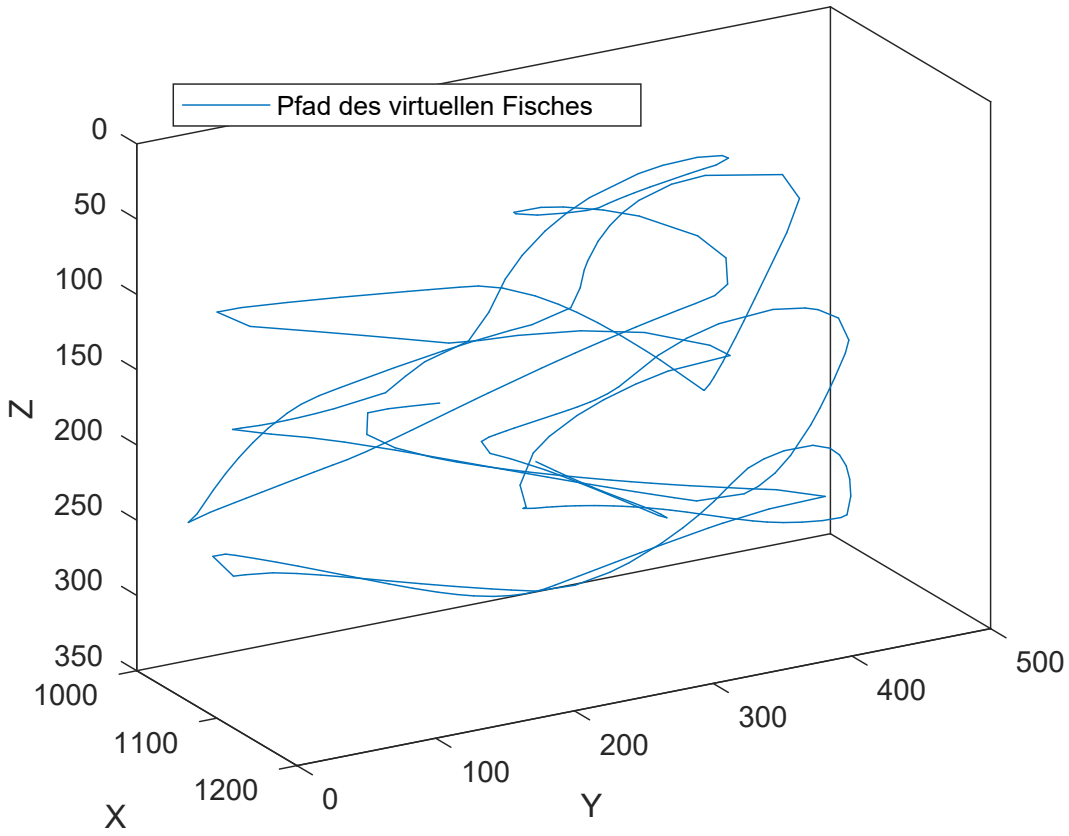


Abbildung 4.10: **Pfade des virtuellen, automatisch animierten Fischstimulus (in mm).** Das Diagramm zeigt einen 60 Sekunden Ausschnitt eines Experiments. Der blaue Pfad entspricht den Steuerdaten des automatischen Animationssystems für den virtuellen Fisch. Der Ursprung des Koordinatensystems liegt in der vorderen, oberen, linken Ecke des realen Aquariums. Da das Virtuelle direkt an das Reale anschließt beginnt der Wertebereich der x-Achse dieser Grafik bei 1000 mm (Breite des realen Aquariums).

Des Weiteren wurde die interaktive Animation im Rahmen des Wissenschaftskommunikationsprojektes "Molly knows best"[4] während der Fußball-Europameisterschaft 2016 eingesetzt. Im Projekt agierte ein weiblicher Breitflossenkärpfling als Fußball-Orakel und konnte im Partnerwahlexperiment zwischen zwei besonders texturierten, interaktiven, virtuellen Stimuli entscheiden. Die Stimuli waren jeweils mit der Landesflagge der gegeneinander antretenden Mannschaften texturiert und "Molly", der lebende Breitflossenkärpfling, konnte durch die Dauer seines Aufenthaltes vor der jeweiligen Animation den vermeintlichen Sieger bestimmen. Während dieser 5-minütigen Versuche, wurde die Aufenthaltszeit automatisch ausgewertet und in der grafischen Oberfläche der Software eingeblendet.

4.7 Fazit

In diesem Kapitel wurde die neue benutzerfreundliche Verfahrenskette zum Design, zur Animation und zur Präsentation von virtuellen Fisch Stimuli für Verhaltensbiologen vorgestellt. Im Folgenden sollen die besonderen Merkmale und Unterschiede der Methode vor dem Hintergrund der bisher vorgestellten Arbeiten hervorgehoben werden.

1. *FishCreator* ermöglicht eine benutzerfreundliche Gestaltung mit einer hohen Variabilität bezüglich Morphologie und Farbgebung. Zudem bietet die Software die Möglichkeit durch die Methode der zufälligen Stimulus-Texturierung den subjektiven menschlichen Einfluss bei der Texturauswahl zu verringern. Auch die stufenlose Größenanpassung des Stimulus eröffnet die Möglichkeit Länge, Breite und Höhe dem jeweiligen Populationsmittelwert anzupassen, wie es Rosenthal vorschlägt [111].

2. Mit *FishSteering* wurde eine neue Methode zur Fischstimulus-Animation mit Hilfe eines Gamecontrollers vorgestellt. Dies ermöglicht Forschern ohne Erfahrung und Kenntnis im Bereich der 3D-Animation Stimulus-Animationen auf einfachem und schnellem Wege zu generieren.

[4]Videos des Orakels sind unter folgendem Link zu finden: `https://www.youtube.com/channel/UCRLgy7c1e7g_6A1LCdXCd7w`

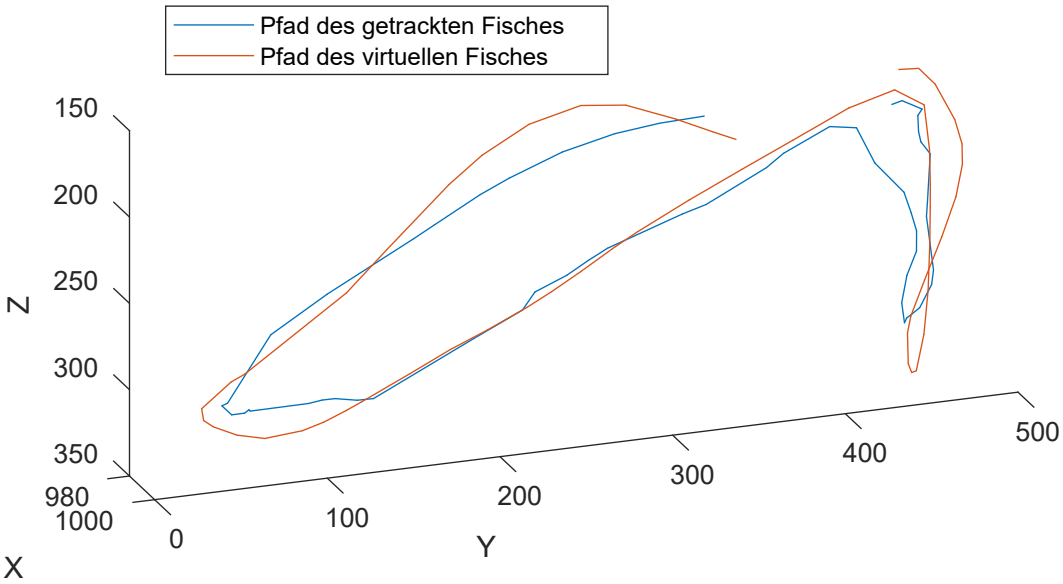


Abbildung 4.11: **Pfade des virtuellen und des realen Fisches während eines interaktiven Versuches (in mm).** Das Diagramm zeigt einen 10 Sekunden Ausschnitt zweier 3D-Fischpfade während eines interaktiven Experiments. Der blaue Pfad des realen Fisches wurde dabei von dem Tracking-System aufgezeichnet; der rote Pfad entspricht den Steuerdaten des Animationssystems für den virtuellen Fisch. Es ist deutlich zu erkennen, wie der virtuelle Fisch dem lebendigen Fisch folgt. Der Ursprung des Koordinatensystems liegt in der vorderen, oberen, linken Ecke des realen Aquariums. Der Pfad liegt im Übergangsbereich des realen zum virtuellen Aquarium (980 mm bis 1020 mm auf der x-Achse).

3. Die aufgenommenen Schwimmpfade können mit beliebigen Stimulusmodellen kombiniert werden (auch mit außergewöhnlichen wie einer Box vgl. [49]) und werden im Gegensatz zu Ingley et al. [54] in Echtzeit gerendert und präsentiert. Dies ermöglicht einen sehr hohen Standardisierungsgrad der Versuche und eine engere Zusammenarbeit zwischen Forschungsgruppen durch die Möglichkeit des Austausches von 3D-Modellen und Schwimmpfaden.

4. Die automatische Animation der Stimuli bietet auch ohne vorherige Aufnahme eines Schwimmpfades eine authentische Stimulus-Animation. Funktionen wie die automatische Bewegung der Rückenflosse oder des Gonopodiums runden die Animation ab und ermöglichen deren Einsatz in Partnerwahlexperimenten.

5. Die interaktive Animation in Kombination mit der in Kapitel 3 entwickelten Methode zum Echtzeit-Fisch-Tracking in 3D ermöglicht es, eines der größten Einschränkungen von virtuellen Stimuli zu überwinden: die fehlende Interaktion zwischen Stimulus und lebendem Fisch. Im Gegensatz zur interaktiven Animation von Butkowski et al. [22], welche auf einem 2D-Tracking-System basierte, bietet die hier vorgestellte Methode erstmals (soweit dem Autor bekannt) die Möglichkeit den virtuellen Stimulus auch im 3D-Raum auf die Bewegungen des lebenden Fisches reagieren zu lassen.

Besonders im Bereich der interaktiven Animation bietet die Methode viele weitere Möglichkeiten, um neue Forschungsfragen zu beantworten und den Einsatz von lebenden Tieren in Versuchen zu minimieren. Als zukünftige Erweiterung des Systems bietet sich eine Aktionserkennung für Fische an. Diese würde weitere Möglichkeiten eröffnen, z. B. eine erweiterte, verhaltensbasierte Interaktion zwischen virtuellen und lebenden Fischen.

Kapitel 5

Motion Capture mittels Analyse-durch-Synthese

Durch die Entwicklung der in Kapitel 4 vorgestellten Methode zur Animation virtueller Fischstimuli ist der Einsatz dieser erheblich vereinfacht worden. Besonders die sonst sehr aufwendige Animation kann mit Hilfe der veröffentlichten Software stark reduziert werden. Allerdings bildet die Animationssoftware nicht alle Bewegungsabläufe ab und spezielle Verhaltensmuster müssen manuell erstellt werden. Zur manuellen Erstellung bieten sich Verfahren wie die → Rotoskopie an, bei der die Projektion des 3D-Modells mittels → Rigging (Animation) Frame für Frame über die Fischaufnahmen des Videos gelegt werden. Dieser Prozess ist sehr zeitaufwendig und wird meistens von erfahrenen Animatoren durchgeführt.
In diesem Kapitel wird eine Methode vorgestellt, mit der es möglich ist, Bewegungsabläufe und Verhaltensmuster von Fischen automatisch aus Videosequenzen zu extrahieren. Die Methode ist in der Lage 3D-Position, Ausrichtung und Knochenstellung eines oder mehrerer zuvor definierter 3D-Modelle aus Videosequenzen abzuleiten. Das Verfahren beruht auf dem modellbasierten Analyse-durch-Synthese Verfahren (siehe [103]). Beim Analyse-durch-Synthese Verfahren wird versucht ein aufgenommenes Bild einer Szene oder eines Objektes mit Hilfe eines 3D-Modells zu synthetisieren. Dies geschieht in einem iterativen Prozess, in dem die Parameter des 3D-Modells solange angepasst werden, bis die synthetisierten Daten eine hohe Ähnlichkeit mit den aufgenommenen Daten erreichen. Durch die hohe Ähnlichkeit des Modells zur Szene bzw. zum

Objekt geht man schlussendlich davon aus, dass die ermittelten Parameter des 3D-Modells auch für die aufgenommene Szene bzw. das Objekt gelten. Somit lassen sich Größe, Position, Ausrichtung oder auch Verformung von Objekten rekonstruieren.

Dieses Verfahren wurde unter anderem zur Posen-Rekonstruktion, zum Tracking von Menschen (vgl. [102]) oder zur Extraktion menschlicher Posen aus einzelnen Bildern (vgl. [62]) eingesetzt. Das Verfahren eignet sich jedoch in besonderer Weise für den hier beschriebenen Anwendungsfall: zum einen entfällt der zeitaufwendige Prozess der Erstellung von 3D-Modellen, wie sie für dieses Verfahren erforderlich sind, da diese bereits durch die Animation vorhanden sind. Zum anderen verfügt der Bewegungsapparat der meisten Fische im Vergleich zum menschlichen Bewegungsapparat über eine kleinere Anzahl an Freiheitsgeraden, was besonders im Hinblick auf die Analyse-durch-Synthese Methode das Risiko eines Konvergenzfehlers reduziert. In der hier vorgestellten Arbeit wird die Methode durch die Synthese der Lichtbrechung, wie sie beim Aquarium auftritt, ergänzt. Außerdem ist das Verfahren in der hier vorgestellten Version in der Lage auch mit partieller Verdeckung von Fischen umzugehen.

Die hier verwendete Analyse-durch-Synthese Methode nutzt zur Rekonstruktion der Fischpose die aus ein oder mehreren Videosequenzen extrahierten Fisch-Silhouetten. Diese werden mit Hilfe von synthetischen Silhouetten, die aus den bereitgestellten 3D-Modellen extrahiert werden, approximiert. Zur Approximation der Modellparameter wird ein Optimierungsverfahren eingesetzt. Beim Einsatz von Optimierungsmethoden ist stets eine gute Initialisierung der Modellparameter von Bedeutung. Für die hier beschriebene Problemstellung wurde ein eigenes Verfahren zur Parameter-Initialisierung entwickelt, welches auf Basis einfacher Merkmale der Fisch-Silhouette arbeitet. Zur Steigerung der Geschwindigkeit wurde das Verfahren zudem um eine Methode erweitert, welche den Merkmalsraum in posenabhängige Untermengen einteilt.

Zur Validierung der Methode wurden Videosequenzen sowohl einer einzelnen als auch zweier Kameras analysiert, die einzelne Fische oder Fischpaare zeigten. 1000 Frames dieser Videosequenzen wurden manuell annotiert und mit den Ergebnissen der präsentierte Methode verglichen.

Teile dieses Kapitels wurden vorab in [86] und [89] veröffentlicht.

5.1 Stand der Technik

Ein Großteil der Motion-Capture-Forschung beschäftigte und beschäftigt sich mit menschlicher Bewegung. Dazu wurden verschiedene Methoden entwickelt: Auf der einen Seite werden tragbare Systeme verwendet, welche am Körper befestigt werden und die Position und Rotation einzelner Körperteile, wie z. B. Kopf, Arm oder Hand aufzeichnen (vgl. z. B. [109]). Auf der anderen Seite werden optische Systeme genutzt. Diese wiederum lassen sich in markerbasierte und markerlose Systeme unterscheiden. Bei den markerlosen Systemen werden sowohl einzelne oder mehrere RGB-Kameras (z. B. [102, 127]) als auch → RGB-D-Sensoren (z. B. [123]) eingesetzt, welche im Besonderen das markerlose Motion Tracking voran gebracht haben. Aus diesen Daten werden zur Rekonstruktion zumeist Bildmerkmale extrahiert: Choi et al.[29] und Hintertoisser et al. [51] setzen auf kanten- oder eckenbasierte Merkmale. Poppe und Poel [104] und Reinbacher et al. [107] nutzen hingegen form- und silhouetten-basierte Merkmale. Aber auch Bildmerkmalspunkte wie *SURF- (Speeded Up Robust Features)*([29]) oder *SIFT-Merkmalspunkte (Scale-invariant feature transform)*[33] und selbst definierte Merkmale wie bei Payet und Todorovic [98] sowie Hintertoisser et al. [51] werden zur Posenrekonstruktion eingesetzt.
Da die meisten Merkmale universell verwendbar und meist nicht für ein spezielles Problem entworfen worden sind, sind einige der Merkmals-Komponenten nicht relevant für die Posenrekonstruktion, nehmen jedoch Rechenzeit in Anspruch. Um die Effektivität und Genauigkeit dieses Prozesses zu steigern, werden in einigen Arbeiten anwendungsbezogene Merkmale ausgewählt und zu einem neuen kompakten Merkmalsvektor zusammengefasst. Die von Chen et al. [27] vorgestellte Methode kombiniert einige Shape-Deskriptoren und selektiert lediglich die Komponenten, welche für das gegebene Problem geeignet sind. In ihrer Arbeit nutzen sie eine Variante des *Adaboost* Algorithmus, um geeignete Deskriptoren zu suchen und diese anschließend in einem kompakten Vektor zu vereinen. Chen et al. [26] erweitern die zuvor vorgestellte Methode durch einen effektiveren Algorithmus zur Merkmalsauswahl. Rasines et al. [106] entwickelten eine Methode, bei der kontur-, polygon-, blob und gradi-

entenbasierte Merkmale für die Handposenerkennung kombiniert werden. Sie wendeten einen sequenziell wachsenden Suchalgorithmus an, um die Genauigkeit der benutzten Klassifikation zu erhöhen und gleichzeitig die Größe des Merkmalsvektors zu verkleinern. Zur Auswertung der Daten und somit zur Bewegungsrekonstruktion werden heute mehr und mehr auf → Deep Learning basierende Methoden eingesetzt, wie z. B. von Cao et al. [25] oder [148] vorgestellt.

Im Gegensatz zur menschlichen Bewegungsrekonstruktion sind bei der Bewegungsrekonstruktion von Fischen einige Einschränkungen zu nennen. Tragbare Motion-Tracking-Systeme sind für Fische nicht verfügbar. Auch markerbasierte Systeme sind nur sehr eingeschränkt einsetzbar, da es zum einen schwierig ist, Marker am Fischkörper zu befestigen und diese zum anderen den Fisch in seiner Bewegung beeinflussen. Auch verfügbare → RGB-D-Sensoren (z. B. Microsoft Kinect) sind für den Einsatz mit Fischen nur eingeschränkt nutzbar. Diese arbeiten meist mit aktivem Licht, welches bei Anwendung mit einem Aquarium (Licht wandert durch verschiedene Medien) zu Reflektionen und Lichtbrechung und somit zur fehlerhaften Tiefendarstellung führt. Aus diesen Gründen setzen die meisten Verfahren zum 3D-Tracking und zur Bewegungsrekonstruktion von Fischen auf Multi-Kamera-Aufbauten. Neben einiger Forschung im Bereich des 2D- und 3D-Fisch-Trackings (siehe Kapitel 3) gibt es nur wenige Arbeiten im Bereich der Bewegungsrekonstruktion bei Fischen. Takahashi et al. [136] stellten eine Methode zur Positions- und Posenrekonstruktion auf Basis zweier orthogonal zueinander aufgenommener Videosequenzen vor. Dabei nutzten sie ein einfaches 3D-Fischmodell, welches auf die einzelnen Frames der Videosequenzen projiziert wurde. Mit Hilfe eines Brute-Force-Suchalgorithmus (→ Brute-Force-Methode) wurden die Modellparameter optimiert bis die Projektion des Modells mit den aufgenommenen Bildern übereinstimmte. Die so generierten Bewegungsdaten wurden genutzt, um ein Fisch-Bewegungsmodell zu erstellen. Butail und Paley [20] schätzten die 3D-Position und die Fischform, um die Bewegung von Schwarmfischen zu analysieren. Sie abstrahierten das Fischmodell mit Hilfe von biegsamen Ellipsoiden. Mit Hilfe dieses Modells und unter Einsatz eines Partikelfilters wurden basierend auf 2D-Silhouetten aus mehreren Ansichten die Position, die Pose und die Biegung der Fische geschätzt. Später wurde diese Methode optimiert und durch ein erweitertes

3D-Modell ergänzt [19]. Im Vergleich zum vorherigen 3D-Modell besteht das erweiterte 3D-Modell aus mehreren Ellipsen, die entlang der dreidimensionalen Fisch-Mittellinie aufgereiht sind und die Biegung des Fisches mit höherer Genauigkeit beschreiben. Zur Annäherung des Modells an die 2D-Silhouetten, nutzten sie ein heuristisches Approximationsverfahren (*Simulated Annealing*). Als Kostenfunktion wurde dabei die Summe der Distanzen zwischen den Konturpunkten und der Modelloberfläche gewählt. Voesenek et al. [146] benutzten ein ähnliches Verfahren, welches jedoch genauer ist und mehr Freiheitsgrade bei der Biegung und Rotation hat. Sie näherten ebenfalls die 2D-Silhouetten an das 3D-Modell, welches aus ineinander übergehenden Ellipsoiden besteht, an. Zur Schätzung der optimalen Modellparameter projizierten sie das Modell auf Basis der Kameraparameter über die Bildaufnahmen. Anhand der Überlappung zwischen projiziertem Modell und den Silhouetten werden die Modellparameter mit Hilfe eines linearen Optimierungsalgorithmus (*Simplex-Algorithmus*) angenähert. Neben der Bewegungsrekonstruktion nutzten sie das System auch, um Kräfte und Momente der Fischbewegung beim Schwimmen abzuleiten.
Im Vergleich zu den bisher vorgestellten Ansätze, unterscheidet sich die in diesem Kapitel beschriebene Methode in folgenden Punkten:

- Bewegungsrekonstruktion für 3D-Fisch-Stimulusanimationen: Die hier beschriebene Methode zur Bewegungsrekonstruktion nutzt als Ausgangsmodell die verfügbaren Animationsmodelle und passt automatisch die Größe an den lebenden Fisch an. Die so gewonnenen Bewegungsdaten können direkt für die Animation genutzt werden.

- Mit der Methode wird die Lichtbrechung während der Bewegungsrekonstruktion synthetisiert, was zu einer höheren Genauigkeit führt.

- Die Methode ist sehr flexibel und kann für einzelne wie mehrere Fische, für Aufnahmen aus einer Kamera oder mehreren Kameras und zur schnellen (ohne Kompensation der Lichtbrechung) oder zur hochpräzisen Bewegungsrekonstruktion eingesetzt werden.

- Die eigens zur Initialisierung der Fischpose entwickelte Methode berechnet auf Basis der Merkmalsentropie posenabhängige Merkmals-Untermengen (*Subsets*), die zur Beschleunigung der Initialisierung führen.

5.2 Architektur des Analyse-durch-Synthese Systems

Ziel des hier beschriebenen Verfahrens ist die Rekonstruktion der Bewegungen eines oder mehrerer Fische mit Hilfe eines oder mehrerer 3D-Stimulus-Modelle, dessen Erstellung in Kapitel 4.2 beschrieben ist. Dazu werden die Modellparameter des 3D-Modells durch einen Optimierungsalgorithmus angepasst, bis sich das 3D-Modell mit dem Fisch in der Videosequenz deckt. Dazu benötigt der Optimierungsalgorithmus eine Fehlerfunktion, die Informationen über die Deckung zwischen Modell und realem Fisch gibt. Die hier verwendete Funktion nutzt zur Berechnung des Fehlers die aus den Videosequenzen extrahierten Fischkonturen. Die Konturen lassen sich robust mit Hilfe des in Kapitel 3.6 beschriebenen Hintergrundsubtraktionsverfahrens aus den Videosequenzen ermitteln. Um diese Konturen mit dem 3D-Modell vergleichen zu können, müssen auch aus dem 3D-Modell Konturen abgeleitet werden. Diese sind stark von der jeweiligen Ansicht bzw. von der Position und Ausrichtung der Kamera abhängig. Um Konturen aus derselben Ansicht zu erhalten, werden die virtuellen Kameras, die zur Erstellung der synthetischen Konturen dienen, auf Basis der zuvor durchgeführten Kalibrierung der realen Kameras (siehe Kapitel 3.4) ausgerichtet. Zur Initialisierung wird ein zusätzliches Verfahren angewendet, welches auf Basis des 3D-Modells eine Merkmalsdatenbank erstellt die später zur schnellen Parameterschätzung dient.

Das hier beschriebene Analyse-durch-Synthese System besteht aus mehreren Komponenten (siehe Abbildung 5.1), die jeweils einen Teilbereich der Methode abdecken. Die Prozesse **Kalibrierung der Kamera** und **Hintergrundsegmentierung** wurden in Kapitel 3 beschrieben und hier nicht näher erläutert. Die **Parameterinitialisierung** ist dem Analyse-durch-Synthese Verfahren vorgelagert und wird in Kapitel 5.4 beschrieben. Die Komponenten zur Bewegungsrekonstruktion bilden den Kern des Systems: Der **Optimierungsalgorithmus** (Kapitel 5.8) nähert die Modellparameter an. Als Basis des **Konturvergleichs** (Kapitel 5.7) dient der ermittelte Fehler zwischen realer und virtueller Kontur. Dieser wird im nächsten Durchlauf von der **Kontur-Extraktion** (Kapitel 5.5) genutzt, um das 3D-Modell entsprechend der Parameter zu bewegen und eine neue Kontur abzuleiten. Dieser Optimierungs-

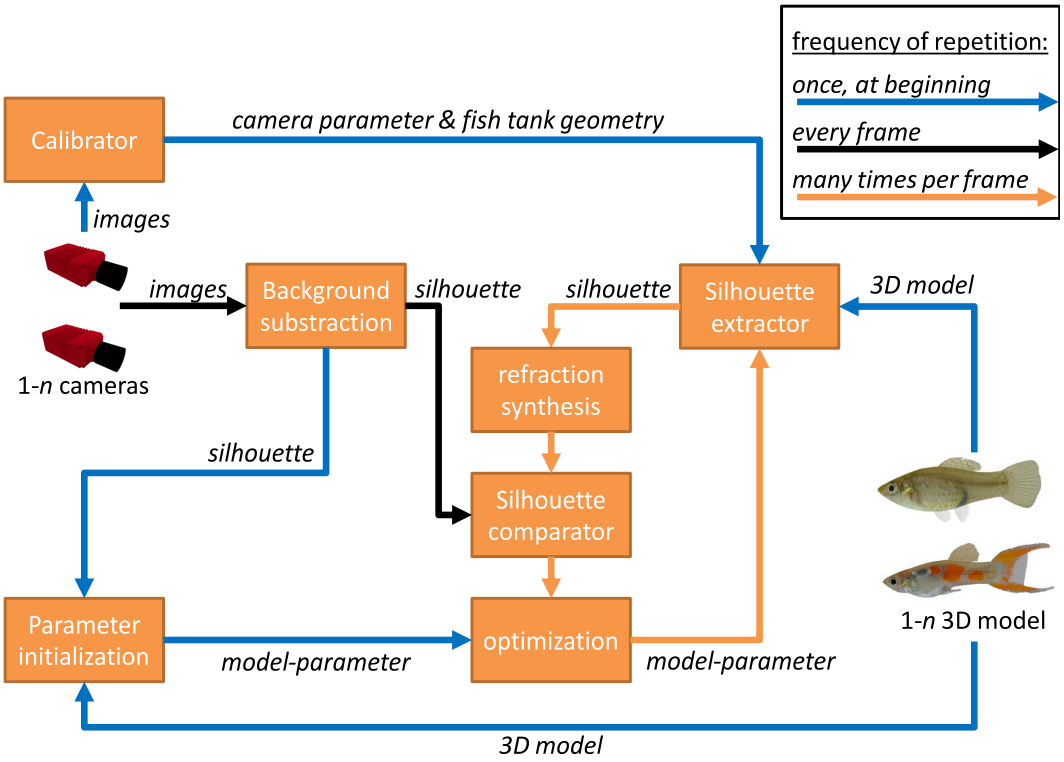


Abbildung 5.1: **Schematische Darstellung des Gesamtsystems.** Die einzelnen Komponenten des Systems sind in orangefarbenen Boxen dargestellt. Die Farben der Verbindungspfeile geben Aufschluss über die Frequenz des Datenaustausches zwischen den Komponenten. Eine erste Version dieser Grafik wurde vorab in [86] veröffentlicht.

durchlauf wird bis zur Unterschreitung eines Fehlerschwellwertes wiederholt. Nach Erreichen des Schwellwertes werden die geschätzten Modellparameter übernommen und dienen als Initialisierungswert für die Posenrekonstruktion im folgenden Frame.

5.3 3D-Modell zur Synthese

Die Qualität der Posenrekonstruktion ist in hohem Maße von der Qualität des 3D-Modells abhängig. Da im hier beschriebenen Fall die Silhouetten des Fisches zur Annäherung genutzt werden, ist besonders die Form des 3D-Modells entscheidend. Diese sollte der Form des lebenden Fisches in großen Teilen gleichen. Die Größe, Breite und Höhe kann dabei vernachlässigt werden, da der Algorithmus vor Beginn des Annäherungsprozesse diese Parameter abschätzt. Je nach Fischart reicht es aus zwei verschiedene Modelle (männlich und weiblich) bereitzustellen. Dies gilt auch für die hier verwendete Art, den Breiflossenkärpfling. Sollte die Art jedoch innerhalb des Geschlechtes in ihrer Form stark variieren, müssen mehrere Modelle bereitgestellt werden, die diese Variationen abdecken.
Grundsätzlich müssen die 3D-Modelle als Mesh-Modelle implementiert sein, um durch den hier verwendeten Algorithmus die Kontur extrahieren zu können. Textur, Färbung und Oberflächeneigenschaften können vernachlässigt werden, da diese für die Kontur unerheblich sind.
Zur Verformungsrekonstruktion muss das Mesh zudem über ein Animationsskelett verfügen, welches entsprechend der Kinematik des Fisches mit dem Mesh verbunden ist. Die Rotationswinkel der Skelettknochen dienen während der Optimierung als Modellparameter.
Die in diesem Beispiel verwendeten 3D-Modelle wurden mit der *FishSim Animation Toolchain* erstellt.

5.4 Initialisierung

5.4.1 Einfache Merkmale zur Initialisierung

Zur schnellen Initialisierung der Fischausrichtung sind insgesamt 21 Merkmale ausgewählt worden, die sich schnell und einfach aus der Kontur und Textur bestimmen lassen. Diese sind in Abbildung 5.2 zu sehen und werden im Folgenden näher beschrieben. Die in den Abbildungen 5.2 d) bis f) genannten Pixelkoordinaten beziehen sich jeweils auf den Schwerpunkt des gedrehten Segmentes.

Winkel zwischen der Segmenthauptachse und der Horizontalen
Mit Hilfe der Bildmomente wird die Hauptmomentenachse des Fisches, der Winkel zwischen dieser und der x-Achse, bestimmt (Abbildung 5.2a).

$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y)$$

$$\alpha = \frac{1}{2} tan^{-1} \frac{2(M_{11} - \overline{x}M_{01})}{M_{20} - \overline{x}M_{10} - M_{02} - \overline{y}M_{01}} \tag{5.1}$$

Verhältnis zwischen Segmentbreite und -höhe
Nach Drehung des Segmentes in die Horizontale um den Winkel α wird das Verhältnis zwischen der Segmentausdehnung in x- und y-Richtung berechnet (Abbildung 5.2b).

Fläche der Segmentviertel
Im ersten Schritt wird der Schwerpunkt des Segmentes mit der Gleichung 3.13 (S. 51) bestimmt. Durch Geraden parallel zur x- und y-Achse wird das Segment in vier Teile geteilt. Die vier Größen der in Abbildung 5.2c eingefärbten Segmentviertel dienten als Merkmal.

Schwerpunkt der Segmentviertel
Mit Hilfe der Gleichung 3.13 werden auch die Schwerpunkte der Segmentviertel bestimmt (Abbildung 5.2d).

Position des Auges
Das Auge der hier verwendeten Breitflossenkärpflinge grenzt sich durch die dunkle bis schwarze Farbgebung gut vom Rest der Textur ab. Mit einer →BLOB-Analyse wird auf der Textur des Segmentes nach dem

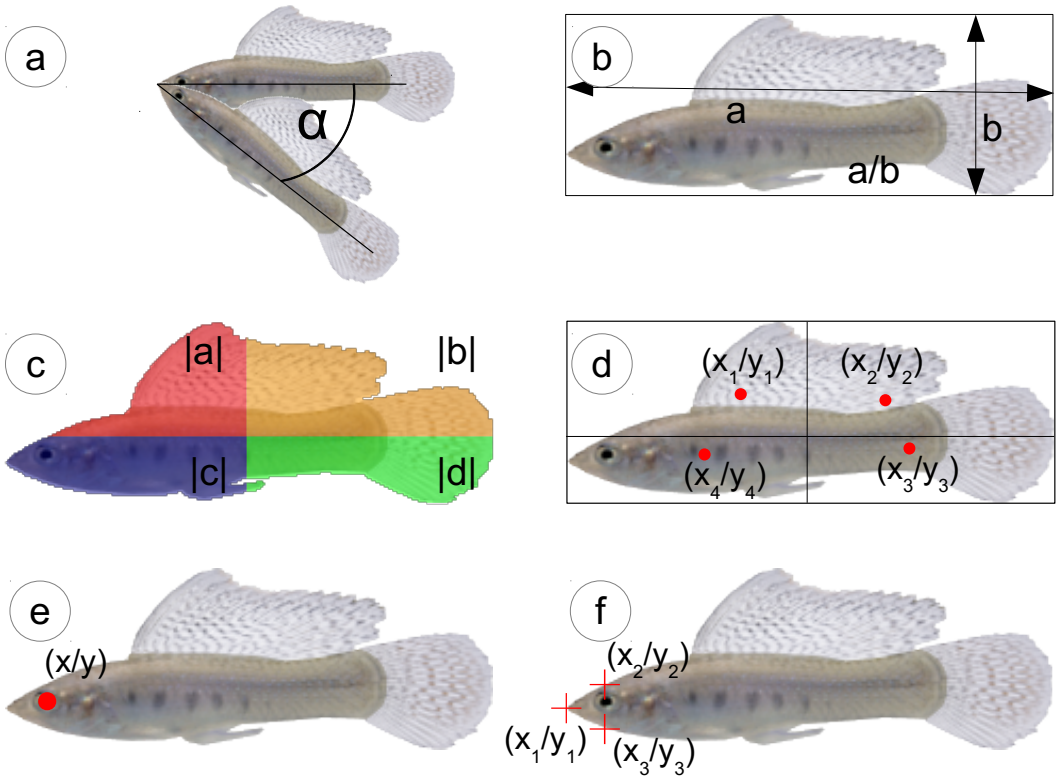


Abbildung 5.2: **Merkmale zur Initialisierung der Pose. (a)** Winkel zwischen der Hauptausbreitungsachse des Segmentes und der Horizontalen **(b)** Verhältnis zwischen Breite und Höhe **(c)** Fläche der Segmentviertel **(d)** Schwerpunkte der Segmentviertel **(e)** Position des Auges **(f)** Pixelposition der Schnauze und der Konturpixel über und unter dem Auge.
Die Grafik wurde vorab in [89] veröffentlicht.

dunklen Auge gesucht. Falls das Auge gefunden wurde, wird sein Schwerpunkt als Merkmal genutzt (Abbildung 5.2e).

Pixelpositionen der Schnauze und der Konturpixel über und unter dem Auge

Als weitere Merkmale dienen die Pixelkoordinaten der Schnauze und der Konturpixel über und unter dem Auge. Zur Bestimmung wird durch die Koordinate des Auges eine Gerade gelegt, die parallel zur y-Achse ist. Die Stellen, an denen die Gerade die Kontur kreuzt, werden als Merkmale übernommen (Abbildung 5.2f).

5.4.2 Auswahl von Merkmalsteilmengen (Feature-Subsets)

Durch die Reduzierung der Merkmalsanzahl lässt sich die Laufzeit der Initialisierung reduzieren. Zum einen verringert sich durch die geringere Anzahl an Merkmalen die benötigte Rechenzeit und zum anderen minimiert sich auch die Laufzeit des späteren Merkmalsabgleiches.
Ziel des hier beschriebenen Verfahrens ist es Merkmalsteilmengen $\tilde{F}_k$ einer Merkmalsmenge F mit den Merkmalen f_1 bis f_n für die 3D-Posenschätzung aus einem 2D-Posenraum (Drehung um die Nick- und Gier-Achse) auszuwählen. Die Auswahl der Merkmalsteilmengen basiert dabei auf der →Entropie der Merkmale im Posenraum. Durch die Drehung des 3D-Modells verändert sich die Entropie der Merkmale. Als Beispiel sei im Falle des Fisch-Modells das Merkmal *Position des Auges* genannt: Dreht sich der Fisch mit dem Schwanz zur Kamera, wird das Fischauge nicht mehr von der Kamera gesehen und dessen Position kann nicht ermittelt werden (siehe Abbildung 5.3). Das Merkmal verliert somit an Bedeutung. Das hier vorgestellte Verfahren sucht auf Basis dieses Effektes posenabhängige Merkmalsteilmengen. Vor jeder Posenschätzung wird auf Basis eines aussagekräftigen Merkmals die passende Merkmalsteilmenge ausgewählt.
Die Methode ist in zwei Teile geteilt: Im ersten Teil wird ein Trainingsdatensatz, der Posen des kompletten Posenraums enthält (360° Drehung um jede Achse in kleinen Schritten von wenigen Grad), analysiert und alle Merkmale berechnet. Die Merkmalswerte werden normalisiert (Kapitel 5.4.2.1), auf den Posenraum projiziert und die Entropie für jedes Merkmal berechnet (Kapitel 5.4.2.2). Anschließend werden auf Basis der berechneten Entropien Merkmalsteilmengen für verschiedene Bereiche des Posenraums definiert (Kapitel 5.4.2.3).

5.4.2.1 Normalisierung der Merkmale

Tests haben gezeigt, dass hohe Rauschwerte eines Merkmals den Entropiewert verfälschen und folglich die Zusammenstellung der Merkmals-Teilmengen negativ beeinflussen. Aus diesem Grund werden die verwendeten Merkmale durch ihren Rauschwert normalisiert.
Zur Berechnung des Merkmalrauschens über den Posenraum wird jedes Merk-

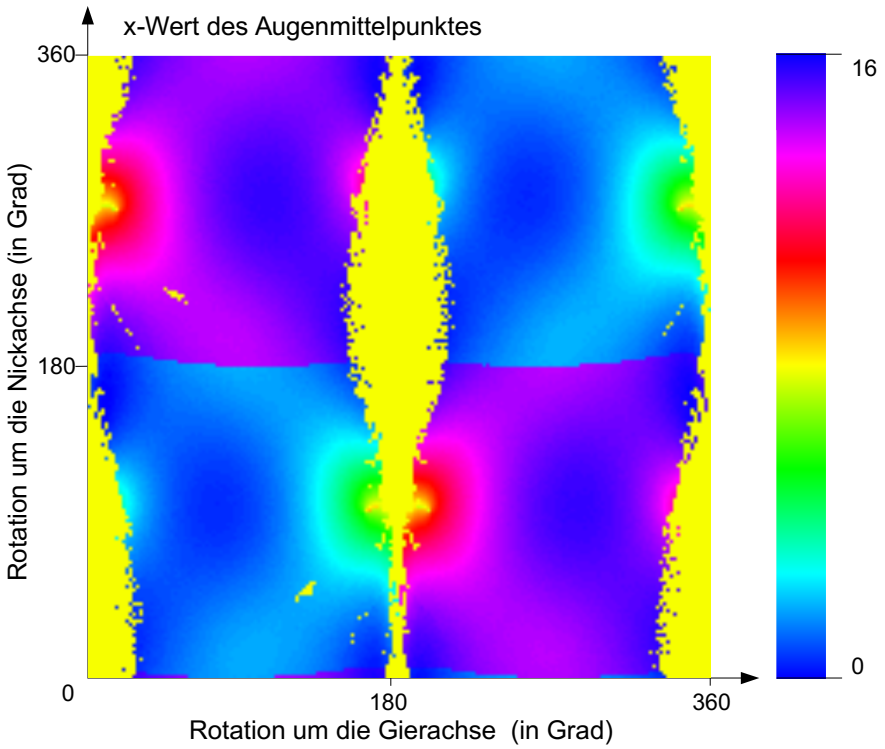


Abbildung 5.3: **2D-Plot der normalisierten Werte des Merkmals *x-Wert des Augenmittelpunktes*.** Der Plot zeigt die Werte des Merkmals über den Posenraum. In einigen Bereichen des Posenraums (im Plot gelb) kann der Algorithmus kein Auge detektieren und der Merkmalswert ist für die entsprechende Pose nicht verfügbar.

mal f_n auf den Posenraum projiziert. Abbildung 5.4 zeigt schematisch den verwendeten zweidimensionalen Posenraum (obere Grafik), der über die Parameter für Drehung um die Gier-Achse (x-Achse in der Abbildung) und Drehung um die Nick-Achse des Fisches (y-Achse in der Abbildung) aufgespannt wird. Durch die Projektion des Merkmals auf den Posenraum ergibt sich für das Merkmal f_n eine zweidimensionale Matrix $M_{fn}(d \times e)$, wobei d die Anzahl der Schritte entlang der x-Achse und e die Schritte entlang der y-Achse beschreibt. $h = d \times e$ entspricht der Gesamtgröße des Trainingsdatensatz. $f_{n_{8,15}}$ beschreibt demzufolge den Wert für das Merkmal f_n bei einer Drehung des Objektes um $8 \times s$ um die Nick-Achse und $15 \times s$ um die Gier-Achse. s beschreibt dabei die Drehwinkel-Schrittweite in Grad:

$$M_{f_n}(d \times e) = \begin{pmatrix} f_{n_{1,1}} & f_{n_{1,2}} & \cdots & f_{n_{1,e}} \\ f_{n_{2,1}} & f_{n_{2,2}} & \cdots & f_{n_{2,e}} \\ \vdots & \vdots & \ddots & \vdots \\ f_{n_{d,1}} & f_{n_{d,2}} & \cdots & f_{n_{d,e}} \end{pmatrix} \tag{5.2}$$

Zur Berechnung des Rauschens wird die Matrix M_{f_n} mit Hilfe eines Boxfilters geglättet und ergibt $\tilde{M}_{f_n}$.

$$\tilde{M}_{f_n}(d, e) = \frac{1}{9} \sum_{u=0}^{2} \sum_{v=0}^{2} M_{f_n}(d - 1 + u, e - 1 + v) \tag{5.3}$$

Nach Subtraktion der Matrix $\tilde{M}_{f_n}$ von M_{f_n} ergibt sich die Rauschmatrix N_{f_n}.

$$N_{f_n} = M_{f_n} - \tilde{M}_{f_n} \tag{5.4}$$

Mit Hilfe des mittleren Rauschwerts

$$\overline{n}_{f_n} = \frac{1}{ef} \cdot \sum_{i=0}^{e} \sum_{j=0}^{f} f_{n_{i,j}} \tag{5.5}$$

kann die Standardabweichung σ_{f_n} bestimmt werden. Das Merkmal wird schlussendlich mit σ_{f_n} normalisiert. Der kleinste Wert des Wertebereiches wird ohne Beschränkung der Allgemeinheit zu Null festgelegt.

$$\hat{f}_{n_{d,e}} = \frac{(f_{n_{d,e}} - min(f_n))}{\sigma_{f_n}}. \tag{5.6}$$

Die normalisierten Merkmale werden in Vektoren gespeichert.

$$\hat{f}_n = \begin{pmatrix} \hat{f}_{n_{0,0}} \\ \hat{f}_{n_{0,1}} \\ \vdots \\ \hat{f}_{n_{d,e}} \end{pmatrix} = \begin{pmatrix} \hat{f}_{n_0} \\ \hat{f}_{n_1} \\ \vdots \\ \hat{f}_{n_h} \end{pmatrix} \tag{5.7}$$

Die Gesamtheit der Merkmale wird in Matrix $\hat{F}$ gespeichert.

$$\hat{F}(h \times n) = \begin{pmatrix} \hat{f}_{1_1} & \hat{f}_{2_1} & \cdots & \hat{f}_{n_1} \\ \hat{f}_{1_2} & \hat{f}_{2_2} & \cdots & \hat{f}_{n_2} \\ \vdots & \vdots & \ddots & \vdots \\ \hat{f}_{1_h} & \hat{f}_{2_h} & \cdots & \hat{f}_{n_h} \end{pmatrix} \tag{5.8}$$

Nach der Normalisierung haben verrauschte Merkmale im Gegensatz zu rauschfreien Merkmalen kleine Zahlenwerte.

5.4.2.2 Berechnung der Merkmalsentropie

Zur Berechnung der Merkmalsentropie wird ein Histogramm eines jeden Merkmalsvektors $\hat{f}_n$ mit k verschiedenen Klassen b_{n_k} erstellt (ein Beispiel ist in Abbildung 5.5 zu sehen). Jede Klasse b_{n_k} hat l_{n_i} Elemente. Die Entropie p_n des Merkmals $\hat{f}_n$ berechnet sich wie folgt:

$$p_n = \sum_{i=0}^{k} \frac{l_{n_i}}{h} log_2(\frac{l_{n_i}}{h}). \tag{5.9}$$

Die Anzahl der Klassen k wird für jedes Merkmal individuell berechnet. Merkmale mit einem hohen Entropie- und geringem Rauschwert werden in viele, Merkmale mit geringerem Entropie- und hohem Rauschwert in wenige Klassen aufgeteilt. Zur Berechnung wird der normalisierte Wertebereich des jeweiligen Merkmals herangezogen. Das Merkmal mit dem größten Wertebereich erhält die höchste Anzahl (c) an unterschiedlichen Klassen. c wird dabei manuell definiert und sollte unter Berücksichtigung der Größe des Trainingsdatensatzes gewählt werden. Im hier gezeigten Beispiel wurde $c \approx \frac{1}{300}h$ gesetzt. Alle

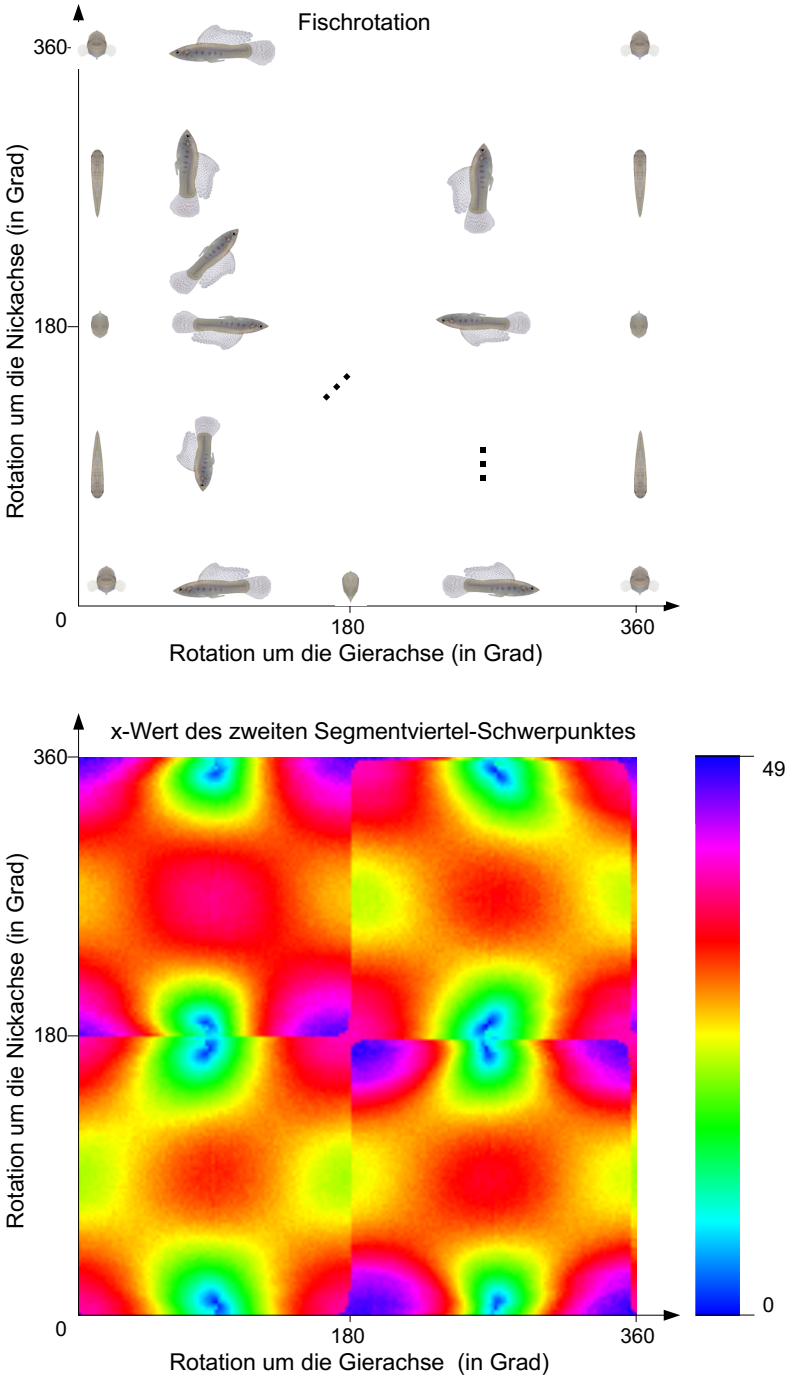


Abbildung 5.4: **Posenraum (oben) und normalisierter Merkmalsvektor (unten).** Die obere Grafik zeigt schematisch den aufgespannten Posenraum. In der unteren Grafik wird ein normalisiertes Merkmal auf den Posenraum projiziert. Eine abgeänderte Version dieser Grafik wurde vorab in [89] veröffentlicht.

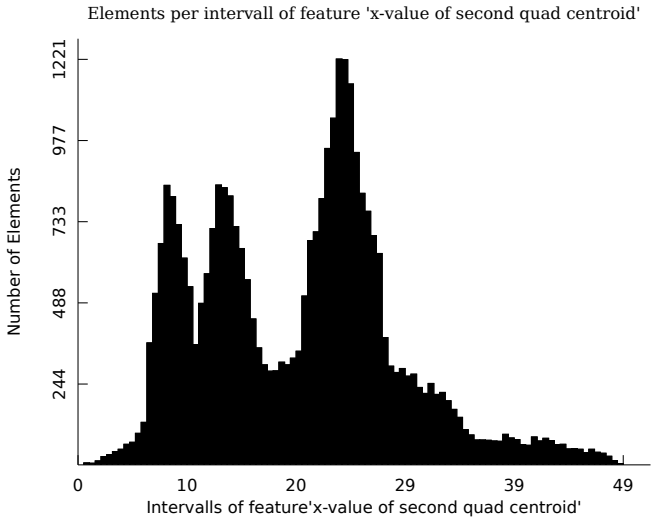


Abbildung 5.5: **Histogramm eines Merkmals.** Das Histogramm stellt beispielhaft die Einteilung eines Merkmals in 50 Klassen dar.

Klassen aller Merkmale haben die gleiche Intervallgröße $|b|$.

$$|b| = \frac{\max\limits_{i=0\ldots n}(\max\limits_{j=0\ldots h}(\hat{f}_{i_j}) - \min\limits_{l=0\ldots h}(\hat{f}_{i_l}))}{c} \qquad mit \;\; \hat{f}_i \in \hat{F} = \begin{pmatrix} \hat{f}_0 & \hat{f}_1 & \ldots & \hat{f}_n \end{pmatrix}. \tag{5.10}$$

Die Anzahl der Klassen k_i des Merkmals $\hat{f}_i$ ergibt sich aus folgender Rechnung

$$k_i = \lceil \frac{\max\limits_{j=0\ldots h}(\hat{f}_{i_j}) - \min\limits_{l=0\ldots h}(\hat{f}_{i_l})}{|b|} \rceil. \tag{5.11}$$

5.4.2.3 Zusammenstellung der Merkmalsteilmengen und Berechnung der Entropie dieser Teilmengen

Zum Zusammenstellen der Merkmalsteilmengen wird im ersten Schritt das Merkmal $\hat{f}_p$ mit der höchsten Entropie gesucht

$$\hat{f}_p = \max_{i=0\ldots n}(p_i). \tag{5.12}$$

Jede Klasse b_{p_k} des Histogramms von $\hat{f}_p$ enthält eine bestimmte Menge von Posensätzen C_{p_k}. Im Umkehrschluss sind alle Posensätze C_{p_k} mit einem ähnlichen

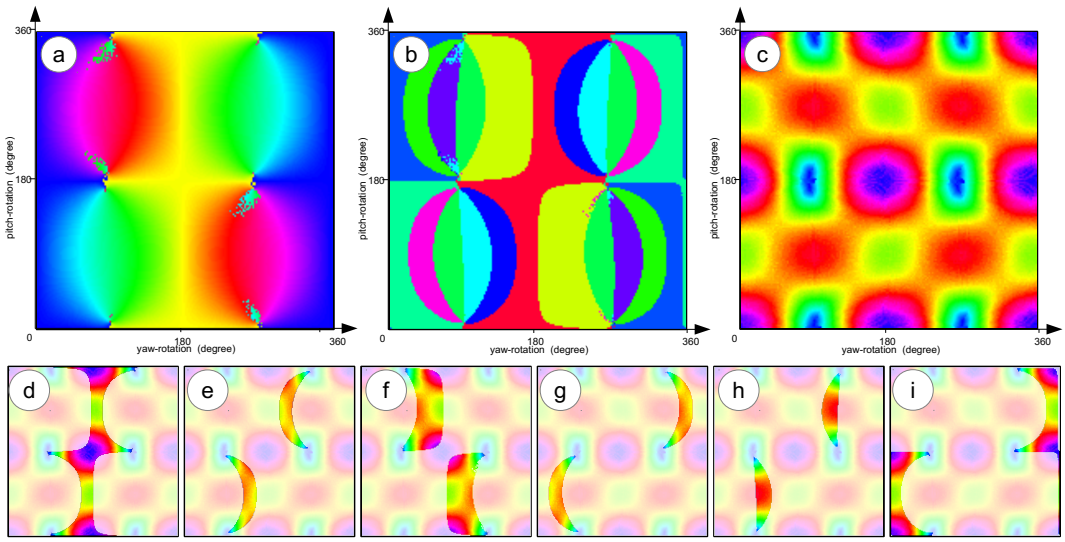


Abbildung 5.6: **Prozess der lokalen Entropieberechnung**
a) Merkmalsprojektion auf den Posenraum (Merkmal: Winkel der Hauptsegmentenachse) **b)** Merkmal aus a) nach Histogrammklassen segmentiert. Jede Farbe entspricht einer anderen Klasse. **c)** Merkmalsprojektion eines zweiten Merkmals **d)-i)** Die segmentierten Bereiche der Klassen des ersten Merkmals werden genutzt, um Bereich des zweiten Merkmals zu segmentieren. Für jeden dieser segmentierten Bereiche wird die Entropie bestimmt. Die Grafik wurde vorab in [89] veröffentlicht.

Merkmalswert für $\hat{f}_p$ in der Klasse b_{p_k} vereint. Abbildung 5.6b zeigt beispielhaft die im Posenraum segmentierten Klassen eines Merkmals. Jede Farbe zeigt eine Klasse b_{p_k} des Merkmals $\hat{f}_p$ an. Im nächsten Schritt erhalten alle Posensätze C_{p_k} eine eigene Merkmalsmatrix $\tilde{F}_c$. Diese Matrix enthält alle Merkmale bis auf das Merkmal $\hat{f}_p$ in dem Intervall von C_{p_k}. Ein Beispiel ist in Abbildung 5.6d)-i) gezeigt. In jedem der gezeigten Bilder d)-i) wird ein Merkmal in den Bereichen der ersten sechs Posensätze $C_{p_{[1\ldots6]}}$ markiert. Anschließend wird mit Hilfe der Gleichung 5.9 für alle Merkmale der Merkmalsmatrix $\tilde{F}_c$ die Entropie bestimmt. Anhand des Entropiewertes werden die Merkmale f_{k_n} in der Matrix sortiert.

$$\begin{gathered}\tilde{F}_c = \begin{pmatrix} f_{c_0} & f_{c_1} & \ldots & f_{c_r} \end{pmatrix} \\ mit\ \ c \in (0, 1 \ldots k_p)\ \ und\ \ p(f_{k_0}) > p(f_{k_1})\end{gathered} \tag{5.13}$$

Die posenabhängigen Merkmalsteilmengen werden nun aus den Matrizen $\tilde{F}_k$ erstellt. Die Anzahl $s \in (0, 1 \ldots n-1)$ der Merkmale pro Teilmenge wird dabei manuell bestimmt: Je größer s ist, desto höher ist der Rechenaufwand und somit die Laufzeit. Die Anzahl der Teilmengen k_p entspricht der Anzahl der Klassen des Merkmals mit der höchsten Entropie $\hat{f}_p$

$$\tilde{F}_k = \begin{pmatrix} f_{c_0} & f_{c_1} & \ldots & f_{c_s} \end{pmatrix} \tag{5.14}$$

5.4.2.4 Auswahl der passenden Merkmalsmenge

Während der Laufzeit muss die passende Merkmalsteilmenge gefunden werden. Dazu wird das im Vorverarbeitungsschritt bestimmte Merkmal mit der höchsten Entropie $\hat{f}_p$ für die ankommende Silhouette berechnet. Der berechnete Entropiewert d wird in die passende Histogrammklasse b_{p_k} des Merkmals $\hat{f}_p$ einsortiert. Die Matrix dieser Klasse bildet die Teilmenge $\tilde{F}_d$.

$$\tilde{F}_d = \tilde{F}_k\ \ falls\ \ d \in b_{p_k} \tag{5.15}$$

5.4.3 Schätzung der initialen Pose und Position

Nach Bestimmung der aktiven Merkmalsteilmenge werden die darin enthaltenden Merkmale aus der aktuellen Silhouette berechnet und normalisiert. Mit Hilfe der → Brute-Force-Methode wird der nächste Nachbar in der Trainingsmenge gesucht und dessen Pose als Initial-Konfiguration angenommen.

Die initiale Position wird ähnlich wie in Kapitel 3.8.2 beschrieben ermittelt: Nach Bestimmung des Silhouetten-Schwerpunktes wird der Strahl des Schwerpunktpixels mit Hilfe der Kamerakalibrierungsmatrix bestimmt. Im Falle eines Mono-Kamera-Aufbaus, bei dem nur eine Silhouette pro Objekt existiert, wird das Objekt auf dem Pixelstrahl des Schwerpunktes in die Mitte des Aquariums geschoben. Im Falle eines Aufbaus mit mehreren Kameras, wird ein grober Schnittpunkt zwischen den Pixelstrahlen der Schwerpunkte berechnet (vgl. Kapitel 3.8.2.2).

5.5 Extraktion der 2D-Silhouetten aus dem 3D-Modell

Grundsätzlich gibt es verschiedene Möglichkeiten die Kontur eines 3D-Modells zu generieren. Die offensichtlichste Methode ist es, das Modell auf eine homogene Fläche zu rendern ($\rightarrow$ Renderer) und dieses anschließend vom Hintergrund zu segmentieren. Die Umrandung des Segmentes entspricht der Kontur. In der Praxis hat sich jedoch gezeigt, dass besonders die $\rightarrow$ Rasterung, die zur Abbildung der Geometrie auf einem pixelbasierten Darstellungsgerät genutzt wird, Probleme bereitet. Die Approximation von Kanten durch das Pixelraster ist ungenau und die Kontur gibt besonders bei kleinen Optimierungsschritten die Änderungen am Modell nicht wieder.

Aus diesem Grund wurde in dieser Arbeit eine klassische Methode zur Silhouetten-Rekonstruktion aus 3D-Meshs genutzt. Sie ist an die in [17] vorgestellte Methode angelehnt. Die Methode basiert auf der Annahme, dass eine Mesh-Kante zu einer Silhouette gehört, wenn eine der beiden anliegenden Dreiecke für die Kamera sichtbar und das andere unsichtbar ist (siehe Abbildung 5.7). Um dieses zu überprüfen, wird im ersten Schritt getestet, ob der Normalenvektor $\vec{n}_k$ des Mesh-Dreiecks $\triangle ABC_k$ in Richtung der Kameraebene zeigt. Dies ist der Fall, wenn das Skalarprodukt der Dreiecksnormale und der Normalen der Kameraebene $\vec{v}$ positiv ist. Die Überprüfung findet für alle k

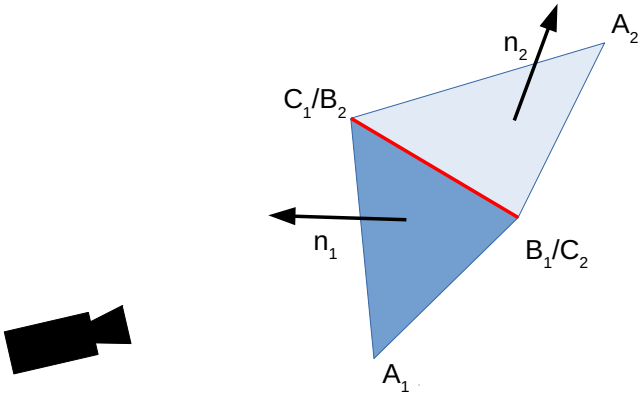


Abbildung 5.7: **Konturextraktion aus dem 3D-Mesh.** Die Grafik zeigt beispielhaft eine Mesh-Kante (rot), die Teil der Silhouette ist. Die zwei angrenzenden Dreiecke sind durch die Eckpunkte A, B, C und durch die Normale $\vec{n}$ beschrieben. Das dunkelblaue Dreieck wird im Gegensatz zum hellblauen von der Kamera gesehen.

Dreiecke des Meshes statt und das Ergebnis wird im Vektor $\vec{f}$ gespeichert.

$$f_i = \begin{cases} 1 & \text{if } \vec{v} \circ \vec{n}_i \geq 0 \quad i \in k \\ 0 & \text{if } \vec{v} \circ \vec{n}_i < 0 \quad i \in k \end{cases} \tag{5.16}$$

Falls f_i und f_j der benachbarten Dreiecke $\triangle ABC_i$ und $\triangle ABC_j$ unterschiedlich sind, teilen sich die beiden eine Silhouetten-Kante. Dieser Test wird für alle benachbarten Dreiecke durchgeführt und im Falle einer gefundenen Silhouettenkante werden die Start- und Endpunkte abgespeichert. Durch die Kamerakalibrierungsmatrix C (siehe Kapitel 3.1.3) werden die 3D-Punkte in 2D-Pixelkoordinaten $\tilde{S}$ umgerechnet.
Je nach Auflösung des 3D-Meshes und je nach Abstand zwischen Kamera und Objekt, kann es vorkommen, dass zwei benachbarte, extrahierte Silhouetten-Pixel mehrere Pixeleinheiten (Silhouetten-Pixel) auseinander liegen. Besonders für den in Kapitel 5.7 beschriebenen Vergleichsalgorithmus, der die nächsten-Nachbarn-Pixel zwischen virtueller und realer Silhouette sucht, kann dies zum Problem werden. Aus diesem Grund werden zwischen zwei benachbarten, virtuellen Silhouetten-Pixeln, die weiter als eine Pixeleinheit auseinander liegen, weitere, durch Interpolation generierte, Silhouetten-Pixel hinzugefügt.

Die hier beschriebene Methode extrahiert aus einem 3D-Mesh-Modell $M(X)$ mit den Modellparametern X den Silhouetten-Pixel-Vektor $\tilde{S}_X$.

5.6 Synthese der Lichtbrechung

Wie in Kapitel 3 gezeigt, kann es bei Kameraaufnahmen von Fischen in Aquarien durch die Lichtbrechung zu Effekten kommen, die bei nicht Beachtung während des Trackings zu Positionsfehlern von einigen Zentimetern führen kann. Besonders bei Systemen, die mit mehreren Kameras arbeiten und Bilder aus verschiedenen Ansichten kombinieren, ist eine Einberechnung der Lichtbrechung unumgänglich.
Beim in Kapitel 3.5 vorgestellten Verfahren zur Lichtbrechungskompensation mittels Strahlverfolgung wird der Lichtstrahl für jedes Pixel beginnend im Kamerazentrum über die Schnitt- und Brechungsebene mit dem Aquarium bis hin zum Fisch Schritt für Schritt verfolgt und berechnet. Bei dem hier vorgestellten Verfahren muss eine andere Strategie angewendet werden, da man umgekehrt bei den Knoten des 3D-Meshes (Kanten der Silhouette) beginnt und den Weg hin zum Kamerazentrum bestimmen muss. Dies wird durch die Lichtbrechung erschwert. Es muss der Durchstoßpunkt mit der Brechungsebene gefunden werden, um den Brechungswinkel berechnen und anschließend den Durchstoßpunkt in der Bildebene bestimmen zu können. Da man jedoch nicht weiß, in welchem Winkel der Lichtstrahl vom Objekt hin zur Brechungsebene abstrahlt, ist auch die Suche nach dem Durchstoßpunkt nicht trivial.
Im hier entwickelten Verfahren wird zur Lösung des Problems im ersten Schritt eine Hilfsebene P aufgespannt. Diese Ebene wird durch den Normalenvektor $\vec{n}$ der Brechungsebene I (je nach Kameraposition ist es die Glasscheibe des Aquariums oder die Wasseroberfläche), dem Ortsvektor der Kamera $\vec{c}$ und dem Ortsvektor der Silhouettenkoordinate $\vec{s_i}$ ($i \in$ allen 3D-Silhouettenkoordinaten). Anschließend wird die Schnittgerade $g(x) : \vec{x} = \vec{o} + x\vec{r}$ zwischen der Hilfsebene P und der Brechungsebene I bestimmt. Die Berechnung dieser Schnittgeraden ist ein Standardproblem der Mathematik und wird hier nicht weiter ausgeführt. Der Durchstoßpunkt mit der Brechungsebene liegt auf der Schnittgeraden $\vec{x}$. Mit Hilfe des Sinussatzes lassen sich Formeln für die Brechungswinkel α und

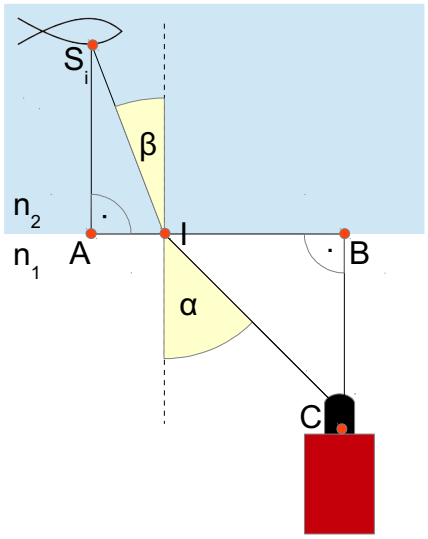


Abbildung 5.8: **Brechung.** Die 3D-Punkte A, B und I liegen in der Brechungsebene. Der Strahl beginnt an der Silhouettenkoordinate S_i, durchstößt die Brechungsebene im Punkt I und trifft im Punkt C die Kamera. Die Grafik wurde vorab in [86] veröffentlicht.

β aufstellen (siehe auch Abbildung 5.8).

$$\frac{n_1}{n_2} = \frac{sin(\beta)}{sin(\alpha)} \tag{5.17}$$

$$sin(\alpha) = \frac{\|(\vec{x} - \vec{c}) \otimes \vec{n}\|}{\|(\vec{x} - \vec{c})\| * \|\vec{n}\|} \tag{5.18}$$

$$sin(\beta) = \frac{\|(\vec{x} - \vec{s_i}) \otimes \vec{n}\|}{\|(\vec{x} - \vec{s_i})\| * \|\vec{n}\|} \tag{5.19}$$

Zur Bestimmung des Schnittpunktes auf der Geraden g, werden die Gleichungen 5.17, 5.18 und 5.19 kombiniert und das daraus resultierende Ergebnis durch eine passende Wahl von x minimiert.

$$\min_{x \in \mathbb{R}} \quad \frac{\|(g(x) - \vec{c}) \times \vec{n}\|}{\|(g(x) - \vec{c})\| \cdot \|\vec{n}\|} \frac{\|(g(x) - \vec{s_i})\| \cdot \|\vec{n}\|}{\|(g(x) - \vec{s_i}) \times \vec{n}\|} - \frac{n_1}{n_2} \tag{5.20}$$

Zur Initialisierung des Minimierungsverfahrens wird für x der Wert x_i angenommen. $g(x_i)$ beschreibt den Schnittpunkt D (Ortsvektor $\vec{d}$) der Geraden zwischen der Silhouettenkoordinate S_i, der Kamera C und der Brechungsebe-

ne I. Der Faktor x_i berechnet sich wie folgt:

$$x_i = \begin{cases} \frac{o_x - d_x}{r_x} \text{ falls } r_x \neq 0 \text{ sonst} \\ \frac{o_y - d_y}{r_y} \text{ falls } r_y \neq 0 \text{ sonst} \\ \frac{o_z - d_z}{r_z} \end{cases} \qquad \text{mit } \vec{d} = \begin{pmatrix} d_x \\ d_y \\ d_z \end{pmatrix}, \vec{o} = \begin{pmatrix} o_x \\ o_y \\ o_z \end{pmatrix}, \vec{r} = \begin{pmatrix} r_x \\ r_y \\ r_z \end{pmatrix} \tag{5.21}$$

5.7 Silhouettenvergleich zur Ähnlichkeitsbestimmung (virtueller/realer Fisch)

Zum Vergleich der Silhouetten des lebenden (S_r) und des synthetischen Fisches S_v wird für jedes Silhouetten-Pixel $v_i \in S_v$ des virtuellen Fisches das nächste Nachbar-Pixel $r_i \in S_r$ aus der Silhouette des lebenden Fisches gesucht. Dazu wird eine $\rightarrow$ Brute-Force-Methode eingesetzt, die auf Basis der Euklidischen Distanz das nächstgelegene Pixel sucht. Die Distanzen zwischen realen und virtuellen Pixeln werden in derselben Reihenfolge wie die Silhouetten-Pixel des lebenden Fisches im Vektor e gespeichert. Dieser Fehlervektor ist abhängig von den Modellparametern X:

$$e(X) = \begin{pmatrix} e_0(X) \\ e_1(X) \\ \vdots \\ e_n(X) \end{pmatrix} = \begin{pmatrix} \|r_0 - v_i\| \\ \|r_1 - v_j\| \\ \vdots \\ \|r_n - v_k\| \end{pmatrix} \quad v_n \in \tilde{S}_X \tag{5.22}$$

Falls der Fisch von mehreren Kameras aufgenommen wird und dementsprechend es mehrere Silhouetten gibt (für jede Ansicht eine), werden die Fehlervektoren der einzelnen Silhouetten in einem gemeinsamen Vektor vereint. $e^k(X)$ beschreibt den Fehlervektor der Kamera k. Der finale Vektor hat dieselbe Größe wie die Summe aller Silhouetten-Pixel aus allen realen Ansichten.

$$e(X) = \begin{pmatrix} e^1(X) \\ e^2(X) \\ \vdots \\ e^k(X) \end{pmatrix} \tag{5.23}$$

5.8 Optimierungsstrategie zur Modellannäherung

Um die synthetisierten Silhouetten an die des lebenden Fisches und somit die 3D-Modellparameter an die Pose des realen Fisches anzunähern, wird in dieser Arbeit ein Optimierungsalgorithmus eingesetzt, der auf der Methode der kleinsten Quadrate basiert. Dieser wurde von Dennis, Gay und Walsh [38] entwickelt und kombiniert die Levenberg-Marquardt-Methode mit einer Quasi-Newton-Methode. Er ist besonders für Probleme geeignet, bei denen die Fehlerfunktion einen größeren Restwert (> 0) hat. Im Falle der hier verwendeten Fehlerfunktion $e(X)$ trifft dieses zu, da eine hundertprozentige Überdeckung der Silhouetten unwahrscheinlich ist.
Während der Optimierung minimiert das Verfahren den Fehlervektor $e(X)$ durch Anpassung der Modellparameter X. Dazu wird die Ableitungsfunktion $D_i(X)$ der Fehlerfunktion genutzt. In dieser Arbeit wird die Ableitungsfunkion für jeden realen Silhouetten-Pixel r_i numerisch approximiert:

$$D_i(X) = \begin{pmatrix} \frac{e_i(X_{0+\epsilon}) - e_i(X_{0-\epsilon})}{2\epsilon} \\ \frac{e_i(X_{1+\epsilon}) - e_i(X_{1-\epsilon})}{2\epsilon} \\ \vdots \\ \frac{e_i(X_{j+\epsilon}) - e_i(X_{j-\epsilon})}{2\epsilon} \end{pmatrix} \; mit \; X_{0+\epsilon} = \begin{pmatrix} x_0 + \epsilon \\ x_1 \\ \vdots \\ x_j \end{pmatrix} \tag{5.24}$$

Sind mehrere Silhouetten aus verschiedenen Ansichten vorhanden, muss der Ableitungsvektor für alle Silhouetten-Pixel erweitert werden. Sollte nur eine Silhouette (einzelne Kamera) verfügbar sein, empfiehlt es sich die Modellparameter zur Stabilisierung zusätzlich mittels Kalman-Filter (siehe Kapitel 3.1.4.5) zu schätzen.

5.9 Schätzung der Fischgröße

Neben der Formähnlichkeit des 3D-Modells muss zur präzisen Schätzung der Modellparameter auch die Größe des 3D-Modells der des lebenden Fisches entsprechen. Eine Möglichkeit ist die manuelle Messung der Fischgröße in Höhe, Breite und Länge. Eine einfachere Möglichkeit besteht darin, das vorhandene Bildverarbeitungssystem zu nutzen. Dazu wird in einem Vorverarbeitungsschritt eine kleine Schwimmsequenz des Fisches analysiert. Dies geschieht ebenfalls mit dem in Kapitel 5.8 vorgestellten Optimierungsverfahren, welches je-

doch nicht nur die Modellparameter zur Position, Ausrichtung und Biegung, sondern zudem die Längenparameter (Breite, Höhe, Länge) des 3D-Modells anpasst. Da es während dieses Prozesses zu kleineren Schwankungen der Größenwerte kommt, werden diese über den Zeitraum der Sequenz gemittelt.
Im Falle mehrerer Kameras (> 1) liefert diese Methode schnelle und präzise Längenwerte. Bei nur einer Kamera kann die Größe nicht ohne weiteres exakt geschätzt werden, da die 2D-Abbildung des Fisches nur in Kombination mit der Entfernung zwischen Kamera und Fisch eine Größenschätzung zulässt. Um trotzdem eine akzeptable Schätzung zu erhalten, wird in dem hier vorgestellten Verfahren eine Optimierung unter Nebenbedingungen durchgeführt. Diese sind durch die Abmessungen des Aquariums gegeben. Die Fischposition wird somit nur innerhalb des Beckens gesucht. Um eine genaue Schätzung zu erhalten, ist es darum wichtig, dass der Fisch in der Videosequenz möglichst die komplette Tiefe des Beckens (aus Sicht der Kamera) durchschwimmt.

5.10 Strategien im Umgang mit mehreren Fischen und Verdeckung

Die vorgestellte Methode ist in der Lage, die Pose mehrerer Fische zu rekonstruieren. Im Falle mehrerer Fische wird jedoch ein Aufbau mit mehreren Kameras empfohlen, um die Stabilität des Algorithmus im Fall einer Verdeckung zu erhöhen.
Solange keine Verdeckung (Fisch - Fisch) besteht, kann das in Kapitel 5.8 vorgestellte Verfahren für jeden Fisch separat angewendet werden. Sollte eine Verdeckung auftreten, wird die Methode wie folgt modifiziert.

Silhouetten-Zuordnung
Wenn nur ein Fisch im Becken schwimmt, ist die Zuordnung der Silhouetten aus verschiedenen Ansichten eindeutig. Falls mehrere Fische im Becken schwimmen, ist die Zuordnung nicht eindeutig. Um trotzdem die aufgenommenen Silhouetten den Fischen zuzuordnen, wird Gleichung 5.22 (S. 163) verwendet. Die aus der Aufnahme extrahierten Silhouetten werden so mit den synthetisierten eines jeden Fischmodells aus dem vorherigen Frame verglichen. Die aufgenommenen Silhouetten werden dem Modell mit dem geringsten Ab-

stand zugewiesen. Dieser Test wird für jede Kamera und zu Beginn jedes Frames durchgeführt.

Silhouetten-Synthese bei Überdeckung
Falls sich Fische in einer Kameraansicht überdecken, liefert die Hintergrund-Segmentierung nur eine Silhouette für die sich verdeckenden Fische. Um diesen Fall auch mit den Modell-Silhouetten darzustellen, werden die Silhouetten jedes beteiligten Fisches separat extrahiert und anschließend zusammengeführt. Übrig bleiben nur die Pixel der Außenkante.

Optimierung bei Verdeckung
Zur Annäherung von vereinten Silhouetten mehrerer Fische wird während der Optimierungsphase der Parametervektor X um die Parameter aller involvierter Fische erweitert. Der Fehlervektor umfasst ebenfalls die Silhouetten-Pixel aller involvierten Fische aus allen Kameraansichten.
Tests haben gezeigt, dass im Falle einer Verdeckung ein höheres Risiko einer fehlerhaften Konvergenz des Optimierungsalgorithmus besteht. Um das Risiko zu verringern, kann in dem Fall, dass nur in einer Ansicht eine Verdeckung besteht, der nicht verdeckten Silhouette eine höhere Bedeutung während der Optimierung zugewiesen werden. Dies geschieht, indem diese Silhouetten-Pixel dem Fehlervektor $e(X)$ doppelt hinzugefügt werden.

5.11 Strategien zum Umgang mit transparenten Teilen des Fisches

Eine besondere Schwierigkeit bei der Fischposen-Rekonstruktion bilden transparente Teile (z. B. Flossen) des Fisches. In den Tests zeigte sich, dass besonders halb-transparente Flossen Probleme bereiten. Diese werden je nach Blickwinkel der Kamera vom Detektionssystem (siehe Kapitel 3.6) erkannt oder ignoriert. Besonders bei dem hier vorgestellten Verfahren ist dies problematisch, da jeweils die Außenkanten des Fischmodells extrahiert werden und diese die Flossen beinhalten. Zur Lösung des Problems wird empfohlen, transparente Teile des Fisches in einer separaten Mesh-Gruppe zu speichern, um diese unabhängig vom Hauptmodell in die Silhouetten-Extraktion einzu-

binden. So ist es möglich, zuerst die Silhouette der nicht-transparenten Teile zu extrahieren und anschließend die Silhouetten der transparenten Teile zu erzeugen. Diese können der Hauptsilhouette hinzugefügt werden. Somit ist sowohl die Außenkante des transparenten als auch des nicht transparent Teils in der Silhouette eingebunden. So kann die Nächste-Nachbar-Suche der Fehlerfunktion (vgl. Kapitel 5.7) entweder die Außenkante der Flosse oder des Fischkörpers finden. Synthetische Konturen mit separater Flossendarstellung sind in Abbildung 5.15 (S. 174) zu sehen.

5.12 Ergebnisse

Neben den Ergebnissen des Analyse-durch-Synthese Verfahrens werden im Folgenden auch die Ergebnisse der zur Initialisierung genutzten Methode der posenabhängigen Teilmengenauswahl beschrieben.

5.12.1 Datensatz

Die hier präsentierten Ergebnisse basieren auf zwei verschiedenen Datensätzen: Der erste Datensatz besteht aus zwei Videosequenzen eines weiblichen Breitflossenkärpflings, der eine Körperlänge von 5 cm (entspricht im Video 180 bis 200 Pixel) aufweist und in zufälligen Bahnen durch ein Aquarium (26 cm x 18 cm x 17 cm) schwimmt. Die Videosequenzen wurden von zwei Kameras (über und vor dem Becken positioniert) aufgenommen und haben eine Länge von 1000 Frames. Die Daten wurden manuell annotiert. Dazu wurde mit Hilfe eines Rendersystems die Projektion des 3D-Modells über die beiden Videosequenzen gelegt und die Modellparameter solange manuell angepasst, bis die Projektionen in beiden Kamerabildern deckungsgleich mit dem Fischbild waren. Das Rendersystem basierte auf einen Ray-Tracing Ansatz und die Lichtbrechung wurde entsprechend des in Kapitel 3.5 vorgestellten Verfahren integriert.
Der zweite Datensatz zeigt zwei weibliche Breitflossenkärpflinge (5 cm Körperlänge, ca. 120 bis 140 Pixel im Bild), die in einem Aquarium der Größe 60 cm x 30 cm x 30 cm dicht nebeneinander das Becken mehrfach durchqueren. Auch dieser Datensatz beinhaltet Videosequenzen zweier Kameras. Die Kameras (Allied Vision Technologies, Prosilica GT1910c) hatten eine Auflösung von

1920 x 1080 Pixel, eine Framerate von 57 Fps und waren synchronisiert.

5.12.2 Modell

Die verwendeten 3D-Modelle von Breitflossenkärpflingen wurden mit der Software *FishCreator* (vgl. Kapitel 4) angefertigt und umfassten 946 Knoten und 36 Knochen. Für den Prozess der Optimierung wurden jedoch nur insgesamt 17 Knochen genutzt: Knochen des Kopfes, vier Wirbelsäulenknochen und zwölf Knochen der Schwanzflosse. Um die Anzahl der Modellparameter zu reduzieren, wurde die in [126] vorgestellte und bereits während der automatischen Animation der Schwimmbewegung genutzte Funktion (siehe Kapitel 4.4.1.1) zur Approximation der Biegung eingesetzt. Dadurch konnte der Parameterraum auf sechs Parameter reduziert werden: Position (x, y und z), Rotation (ψ und θ) und Biegung. Während der Initialisierung wurde der Parametersatz um drei Größenparameter des Fischmodells erweitert.

5.12.3 Initialisierung auf Basis von Merkmalsteilmengen

In einem Vorverarbeitungsschritt wurden 32400 synthetische Bilder aus einem 3D-Fischmodell eines männlichen Breitflossenkärpflings erzeugt (Näheres zur Modellerzeugung in Kapitel 4), die diesen in 2°-Schritten um die Nick- und Gierachse gedreht zeigen. Die Bilder hatten eine Auflösung von 400 x 500 Pixel. Aus diesen wurden die Merkmale extrahiert und posenabhängige Merkmalsteilmengen bestimmt (siehe Abbildung 5.9). Dabei wurde die Anzahl der Merkmale pro Teilmenge variiert. Es wurden Teilmengen mit 5, 10, 15 und 20 Merkmalen erzeugt.

Zum Test der merkmalsbasierten, initialen Posenschätzung wurden 1000 Bilder, die den Fisch in zufällig gewählten Rotationen um die Gier- und Nickachse zeigten, aus dem 3D-Modell abgeleitet. Zur Bestimmung der Genauigkeit des Verfahrens wurde die Differenz zwischen geschätztem Gier- und Nickwinkel und den Groundtruth-Daten berechnet. Dieser Vorgang wurde für Teilmengen mit unterschiedlicher Größe wiederholt. Da die Trainingsdaten die Posen nur in 2°-Schritten abbilden, ist die erwartete mittlere Abweichung mit 0,5° anzunehmen. Zudem wurde die durchschnittliche Abweichung der Rotationen durch Ausreißer negativ beeinflusst. Das liegt daran, dass die Posen im Trai-

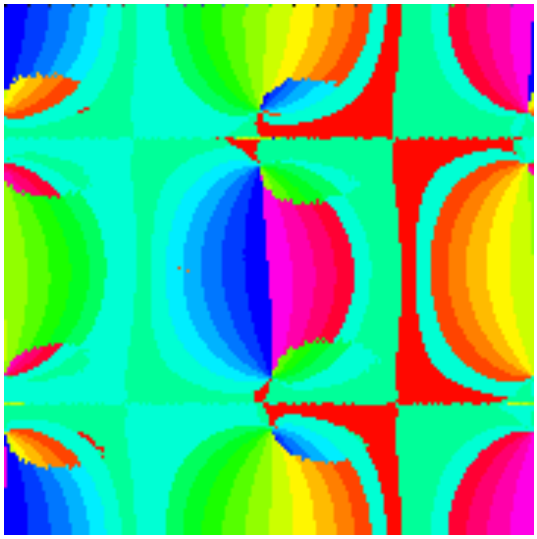


Abbildung 5.9: **Verteilung der Merkmalsteilmengen über den Posenraum.** Jede Farbe der Abbildung repräsentiert eine andere Teilmenge des Merkmals. Im hier gezeigten Fall mit fünf Merkmalen pro Teilmenge ergeben sich für den gesamten Posenraum 22 unterschiedliche Teilmengen. Die Grafik wurde vorab in [89] veröffentlicht.

ningsset 180° um Gier- und Nickachse abdecken und der Fisch von oben und unten sehr große Ähnlichkeit aufweist. Im Fall der konkreten Anwendung sollte der Trainingssatz auf die der Art entsprechenden Posen verringert werden. Zur Reduzierung des Einflusses von Ausreißern im hier durchgeführten Test wurde neben dem einfachen Mittelwert auch ein Mittelwert aus dem Besten der fünf Nächsten-Nachbarn gebildet (siehe Abbildung 5.10).

5.12.4 Kompensation der Lichtbrechung

Die Methode zur Synthese der Lichtbrechung wurde auf beide Datensätze angewendet. In beiden Anwendungen konnte eine deutliche Verbesserung besonders in den Randbereichen des Aquariums festgestellt werden. Dies zeigt sich im Vergleich mit den manuell ermittelten Parameterwerten: Der durchschnittliche Positionsfehler betrug bei der Methode mit aktivierter Brechungssynthese 1,13 mm (Standardabweichung 1 mm, höchste Abweichung 3,9 mm). Ohne Berücksichtigung der Lichtbrechung betrug die mittlere Abweichung 8,24 mm

Abbildung 5.10: **Mittlere Posenabweichung.** Das obere Diagramm zeigt die mittlere Abweichung unter Verwendung von 5, 10, 15 oder allen 24 Merkmalen. Beim unteren Diagramm wurde der Mittelwert über die Beste der fünf Nächsten-Nachbarposen der Trainingsdaten gewählt. Die Grafik wurde vorab in [89] veröffentlicht.

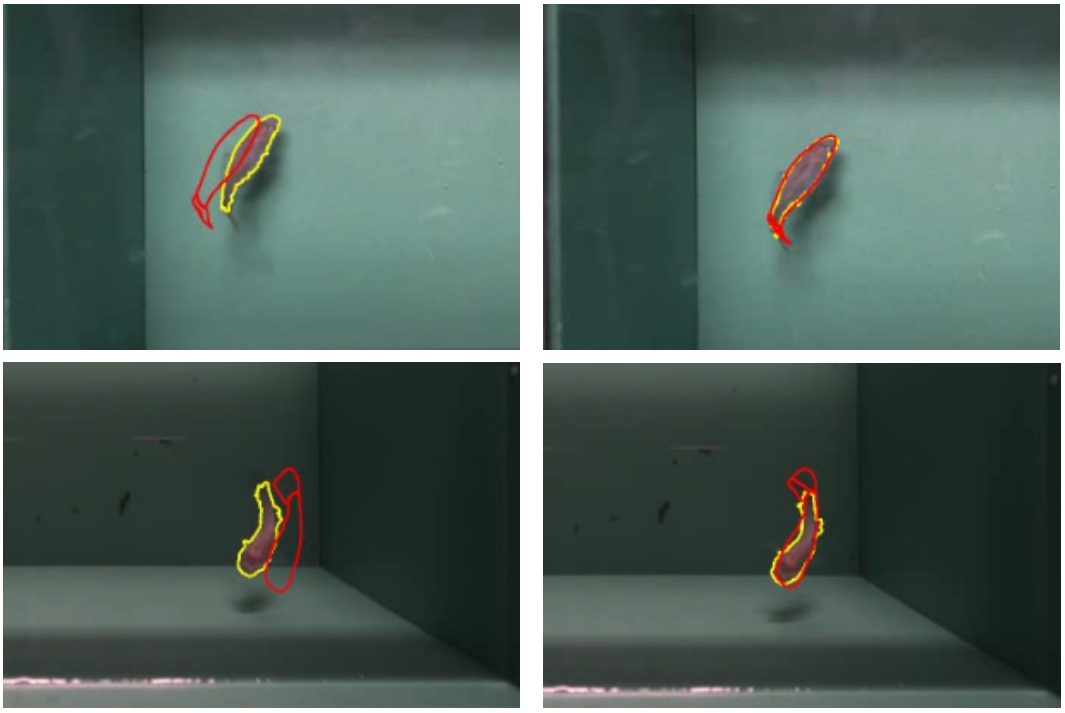


Abbildung 5.11: **Positionsabweichung durch Lichtbrechung.** Die Bilder auf der linken Seite zeigen die Posenrekonstruktion ohne Berücksichtigung der Lichtbrechung. Bei den Bildern auf der rechten Seite wurde die Synthese der Lichtbrechung mit einbezogen.

(Standardabweichung 4,91 mm, höchste Abweichung 19,5 mm). Eine beispielhafte Darstellung der Unterschiede ist in Abbildung 5.11 zu sehen. Auch bei den Rotations- und Biegeparametern lieferte die Methode mit Brechungssynthese durchgehend bessere Ergebnisse. Dies ist damit zu begründen, dass die Silhouetten bedingt durch die Lichtbrechung nicht in beiden Ansichten zur perfekten Deckung gebracht werden können. Weitere Ergebnisse sind in den Abbildungen 5.12, 5.13 und 5.14 zu finden.

5.12.5 Rekonstruktion aus einer und mehreren Ansichten

Die Methode kann mit beliebig vielen Silhouetten aus unterschiedlichen Ansichten durchgeführt werden. Im Rahmen dieser Arbeit wurden die Ergebnisse unter Anwendung einer und zweier Kameras gegenübergestellt. Wie zu erwarten waren die Ergebnisse unter Verwendung zweier verschiedener Ansichten

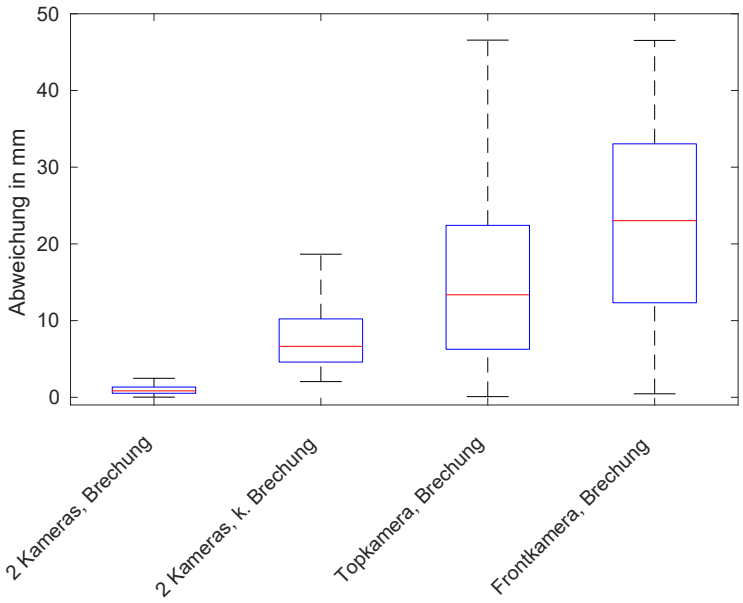


Abbildung 5.12: **Positionsfehler.** Das Diagramm zeigt die Positionsabweichung in Abhängigkeit von der Methode an. Die roten Linien geben jeweils den Mittelwert an, die obere und untere Kastenbegrenzung das 25 %- und das 75 %-Perzentil. Die Antennen geben die größte Abweichung an, wobei keine Ausreißer berücksichtigt werden. Die Grafik wurde vorab in [86] veröffentlicht.

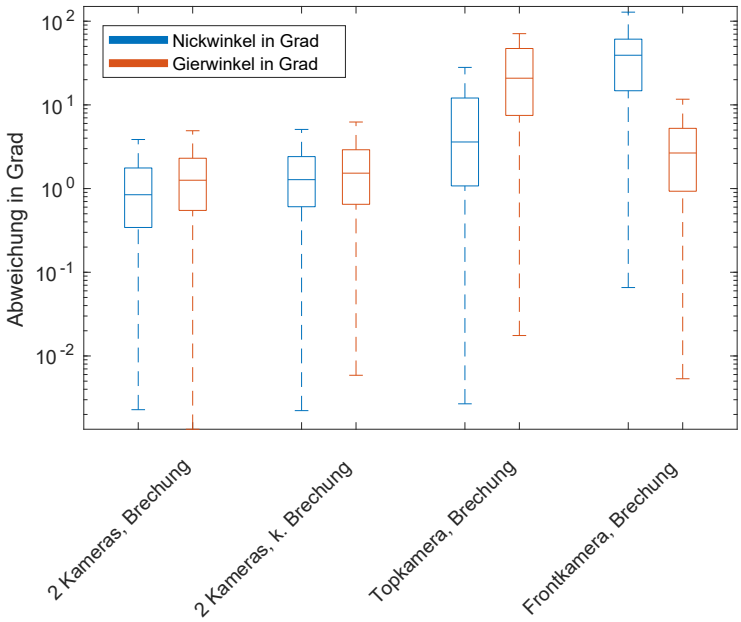


Abbildung 5.13: **Rotationsfehler.** Das Diagramm zeigt die Rotationsabweichung in Abhängigkeit von der Methode an. Die Grafik wurde vorab in [86] veröffentlicht.

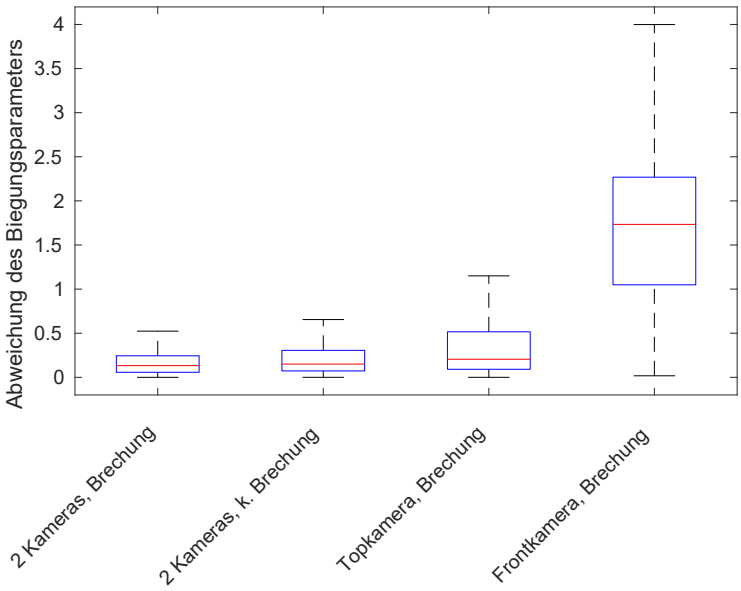


Abbildung 5.14: **Abweichung des Biegungsparameters.** Das Diagramm zeigt die Abweichung zwischen annotiertem und berechnetem Biegeparameter aufgeteilt nach Methoden. Die Grafik wurde vorab in [86] veröffentlicht.

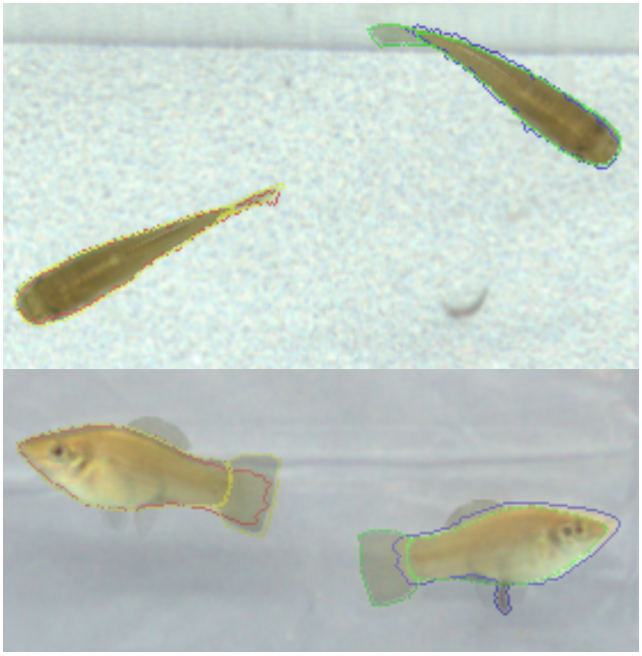


Abbildung 5.15: **Overlay der synthetischen und extrahierten Silhouetten.** Die grünen und gelben Umrandungen stellen die synthetischen, die roten und blauen die extrahierten Silhouetten dar. Die Grafik wurde vorab in [86] veröffentlicht.

besser (siehe Abbildungen 5.12, 5.13 und 5.14). Unter Nutzung einer Kamera erhöhte sich besonders der Fehler der Dimension, die in Blickrichtung der Kamera lag. Im hier gezeigten Beispiel mit Breitflossenkärpflingen konnten bessere Ergebnisse erzielt werden, wenn die Topkamera anstatt der Frontkamera verwendet wurde. Dies liegt insbesondere daran, dass die Topkamera auch die Biegung des Fisches erfasst. Für Experimente mit eingeschränkter dritter Dimension (z. B. im Falle eines niedrigen Wasserstandes) kann ein Aufbau mit einer Kamera eine kostengünstige Alternative darstellen.

5.12.6 Verdeckung

Aus dem zweiten Datensatz wurden die Posen, Positionen und Biegungen zweier nebeneinander schwimmender Fische aus zwei verschiedenen Ansichten rekonstruiert. Dabei verdeckten sich die Fische mehrmals gegenseitig. Der Algorithmus lieferte zumeist zuverlässige Posen (siehe Abbildung 5.16). Im Falle

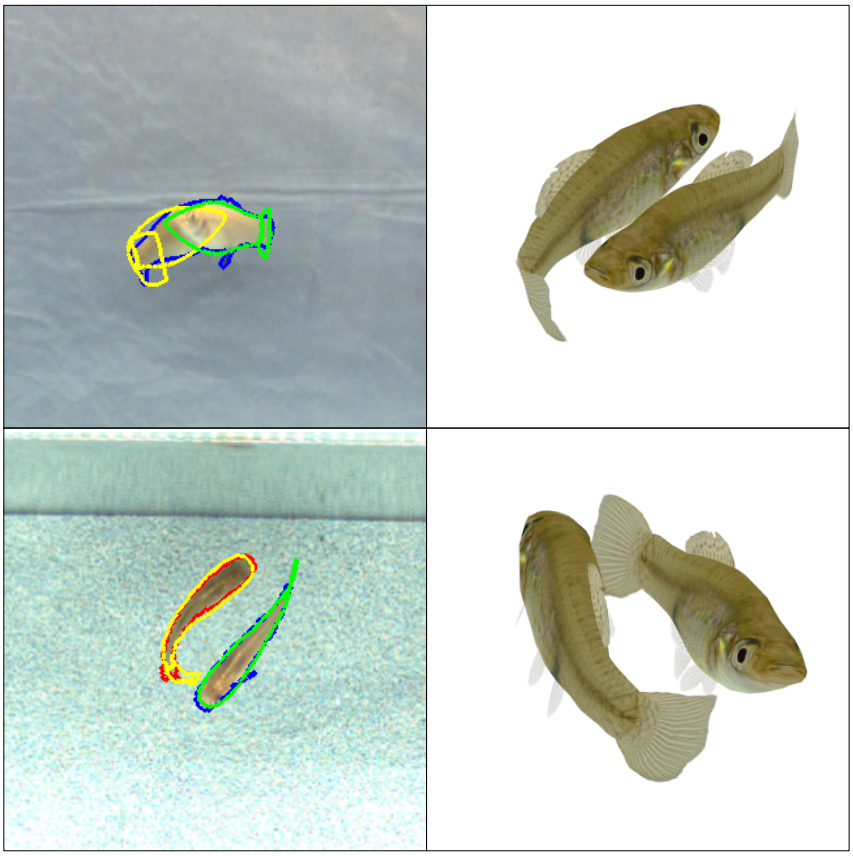


Abbildung 5.16: **Rekonstruktion zweier Fischposen.** Die grünen und gelben Umrandungen in den linken Grafiken stellen die synthetischen, die roten und blauen die extrahierten Silhouetten dar. Die rechten zwei Grafiken zeigen die gerenderte Rekonstruktion aus zwei verschiedenen Ansichten. Die Grafik wurde vorab in [86] veröffentlicht.

einer Verdeckung in beiden Ansichten kam es vereinzelt zu kurzzeitiger Abweichung (siehe Abbildung 5.17). Durch Verwendung weiterer Kameras kann die Gefahr einer falschen Konvergenz reduziert werden. Des Weiteren bietet sich auch die Verwendung eines Kalman- oder Partikelfilters an, um kurze Ausfälle zu kompensieren.

5.12.7 Laufzeit

Die Software wurde auf einem System mit Intel I7-3770 CPU (4 x 3,4 GHz), 16 GB Arbeitsspeicher und Ubuntu 14.04 Betriebssystem getestet.

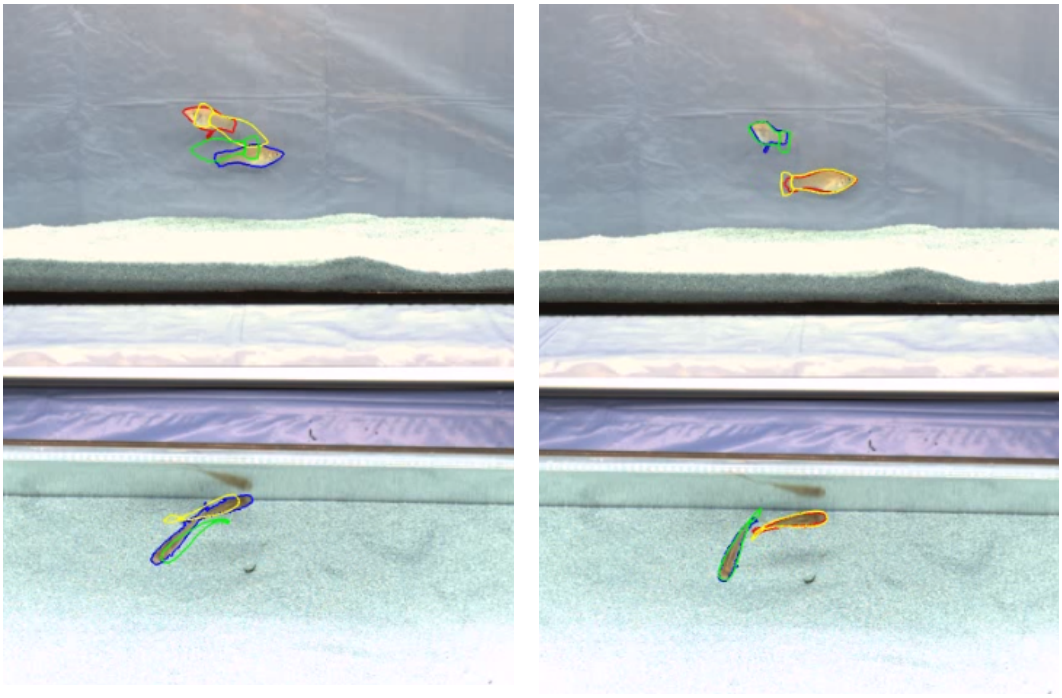


Abbildung 5.17: **Fehler bei der Optimierung.** Durch eine kurzzeitige Verdeckung in beiden Ansichten kommt es zu Abweichungen bei den Modellparametern (linke Bilder). Die extrahierten Silhouetten (gelb und grün) sind nicht mit denen der Fische (blau und rot) deckungsgleich. Dieser Zustand ändert sich zumeist nach kurzer Zeit (rechtes Bild).

Das Verfahren zur Initialisierung und Auswahl der Teilmengen wurde in *c++* implementiert. Während der Initialisierung, konnte durch den Einsatz der posenabhängigen Merkmalsteilmengen die Laufzeit in zwei verschiedenen Prozessstufen reduziert werden: Zum einen konnte die Zeit zur Merkmalsextraktion durch die Reduzierung der Merkmalsanzahl um bis zu 60 % gesenkt werden (siehe Abbildung 5.18 oben). Zum anderen konnte auch die Laufzeit der Brute-Force-basierte Suche in den Trainingsposen um bis zu 41 % gesenkt werden (sieh Abbildung 5.18 unten). Die Laufzeit wurde für verschiedene Größen der Merkmals-Teilmengen berechnet. Da einige der Merkmale voneinander abhängen (siehe auch Abbildung 5.2), ist die Laufzeit der Merkmalsextraktion in den Tests ab einer Anzahl von 15 Merkmalen pro Teilmenge gleich.

Zur Implementierung des Analyse-durch-Synthese Verfahrens, wurde zum einen die Bildverarbeitungsbibliothek *OpenCV* (Version 2.4.12, `http://opencv.org`) und zum anderen die Softwarebibliothek *Dlib* (Version 19.2, `http://dlib.net`), welche zur Optimierung eingesetzt wurde, benutzt. Die Extraktion der Silhouette wurde mit Hilfe der → Spiel-Engine *irrlicht* (Version 1.81, `http://irrlicht.sourceforge.net/`) durchgeführt. Zur Laufzeitmessung wurde die Zeit gemessen, die benötigt wurde, um die Posen aller Fische aus einem Frame (im Falle von zwei Kameras, aus zwei Frames) zu extrahieren. Tabelle 5.1 gibt einen Überblick über die Laufzeit in Abhängigkeit von der Konfiguration an. Die Synthese der Lichtbrechung ist aufgrund der Optimierung jedes einzelnen Pixelstrahls (vgl. Kapitel 5.6) sehr rechenaufwendig. Durch eine Substitution der Optimierungsgleichung 5.20 mit einer analytische Funktion, könnte die Rechenzeit der Synthese zukünftig reduziert werden. Grundsätzlich kann festgehalten werden, dass die Laufzeit mit der Anzahl der Fische und Kameras steigt. Durch den Einsatz aktueller Computerhardware ist der Algorithmus auch in Echtzeit einzusetzen.

5.13 Fazit

In diesem Kapitel wurde ein Verfahren zur automatischen Posen und Bewegungsrekonstruktion von Fischen mittels 3D-Animationsmodell vorgestellt. Das Modell beinhaltet mehrere neue Ansätze. Zunächst wurde zur Initialisierung des Verfahrens eine Methode vorgestellt, mit deren Hilfe poseabhängige

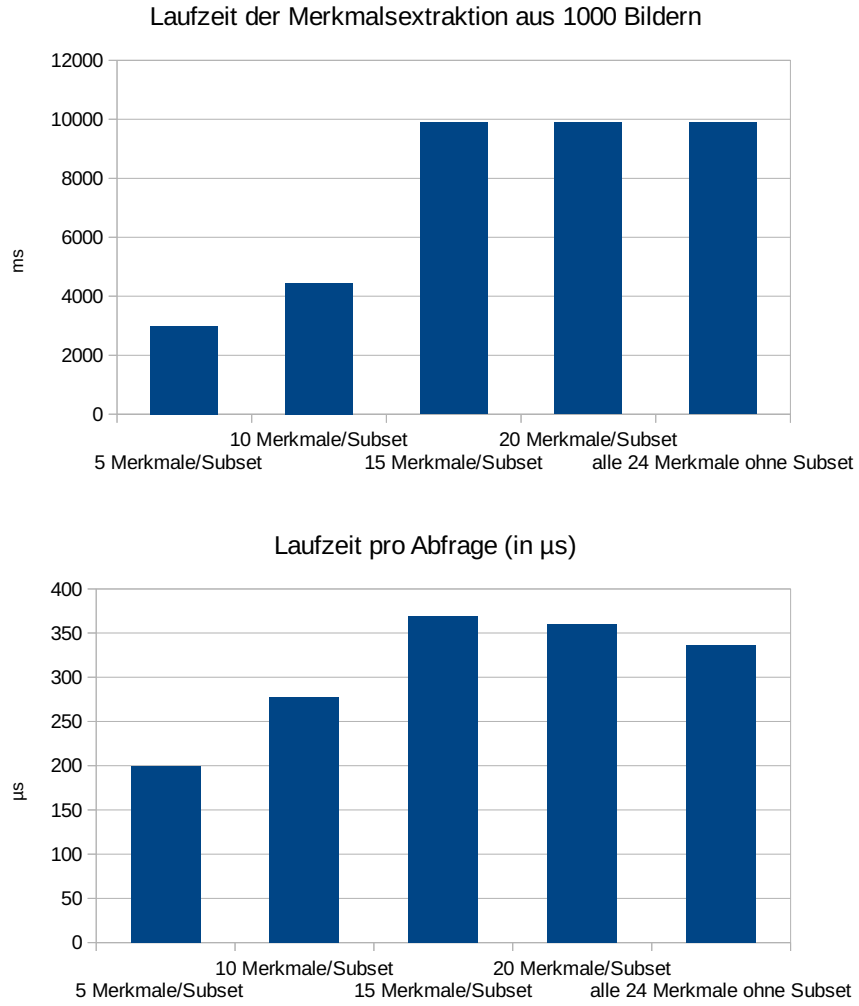


Abbildung 5.18: **Laufzeit der Merkmalsextraktion und des Merkmalsabgleiches.** Das obere Balkendiagramm stellt die Laufzeit dar, die benötigt wird, um aus 1000 Bildern, die für die jeweilige Teilmenge benötigten Merkmale (fünf bis 20 Merkmale), zu extrahieren. Zudem wird auch die Laufzeit der Berechnung aller 24 Merkmale angezeigt. Das untere Diagramm gibt die Zeit zur Suche des nächsten Nachbarn in den Trainingsdaten in Abhängigkeit von den Teilmengengrößen an. Die Grafik wurde vorab in [89] veröffentlicht.

Tabelle 5.1: Laufzeit des Algorithmus mit unterschiedlichen Konfigurationen

Konfiguration	**Laufzeit pro Bildaufnahme in Sekunden**
ein Fisch, eine Kamera, keine Synthese der Lichtbrechung	0.07
ein Fisch, eine Kamera, Synthese der Lichtbrechung	0.25
ein Fisch, zwei Kameras, keine Synthese der Lichtbrechung	0.14
ein Fisch, zwei Kameras, Synthese der Lichtbrechung	0.5
zwei Fische, zwei Kameras, Synthese der Lichtbrechung	0.7
ein Fisch, zwei Kameras, Synthese der Lichtbrechung und Größenschätzung des Fisches (drei zusätzliche Parameter)	0.67
ein Fisch, zwei Kameras, Synthese der Lichtbrechung und acht zusätzliche Biegungsparameter	0.92
ein Fisch, zwei Kameras, keine Synthese der Lichtbrechung und acht zusätzliche Biegungsparameter	0.25

Merkmalsteilmengen bestimmt werden, so dass die Laufzeit der initialen Posenschätzung stark reduziert werden kann. Gleichzeitig konnte mit Hilfe der Merkmalsteilmengen die Genauigkeit der Initialisierung erhöht werden.
Zumdem wurde eine neue Methode zur Synthese der Lichtbrechung im Analyse-durch-Synthese Verfahren entwickelt. Es wurde gezeigt, dass so eine Reduzierung des Positionsfehlers um bis zu 85 %, des Rotationsfehlers um bis zu 20 % und der Biegungsabweichung um bis zu 11 % möglich ist. Des Weiteren wurde das Analyse-durch-Synthese Verfahren durch weitere fischspezifische Funktionen verbessert: Neben einem Verfahren zur Bewältigung des Problems der Verdeckung, wurden Strategien zum Umgang mit transparenten Körperteilen wie Flossen entwickelt.
Das hier vorgestellte Analyse-durch-Synthese Verfahren eröffnet die Möglichkeit, automatisch Bewegungsmuster aus Videosequenzen mit 3D-Animationsmodellen zu extrahieren und diese nahtlos in eine 3D-Animation zu überführen. Für zukünftige Projekte ist auch eine Echtzeitanwendung des Verfahrens denkbar und somit eine genaue Bewegungsanalyse des Fisches möglich. Dies würde die Tür für eine weiterführende, interaktive Fischsimulation öffnen: Durch die genaue Rekonstruktion der Fischpose können weitere Verhaltensmuster erkannt und die Stimulus-Animation gezielter eingesetzt werden.

Kapitel 6

Zusammenfassung und Ausblick

Der im Jahre 1950 von Alan Turing entwickelte gleichnamige Test bescheinigt einem Computer künstliche Intelligenz, wenn ein Mensch innerhalb einer Chatkommunikation mit diesem nicht unterscheiden kann, ob der Kommunikationspartner ebenfalls ein Mensch oder eine Maschine ist. In der hier vorgestellten Arbeit wurde ein System für die Verhaltensforschung entwickelt, welches nicht einen Menschen sondern einen Fisch täuschen kann und somit den "Turing-Test" für Fische bestanden hätte.
Das System umfasst zahlreiche Ansätze und Methoden zur Erstellung virtueller Fischstimuli und für deren Einsatz in der Verhaltensforschung. So eröffnen sich für diese Forschungsdisziplin neue Möglichkeiten und Chancen hinsichtlich der Durchführbarkeit und Reproduzierbarkeit verhaltensbiologischer Experimente.
Ziel der Arbeit war die Erstellung und Animation eines 3D-Fischstimulus, welcher in der Lage ist mit einem reellen Fisch in Echtzeit zu interagieren. Zur Erstellung der Fischanimationen wurde eine Verfahrenskette entwickelt und getestet, die dem Benutzer bei der Erstellung und Animation von 3D-Fisch-Modellen unterstützen. Der Schwerpunkt lag dabei auf der Entwicklung von Verfahren, die auch unerfahrenen Benutzern eine einfache und benutzerfreundliche Animation ermöglicht. Kernkomponente des Systems ist die halbautomatische Animation des 3D-Modells mittels Gamepad, die automatische, artspezifische Animation und die interaktive Animation, welche in Kombination mit dem Tracking-System eine Kommunikation zwischen realem und virtuellem Fisch ermöglicht. Eine der zentralen Herausforderungen dieser Arbeit

aus Sicht der Bildverarbeitung war der Umgang mit der am Aquarium auftretenden Lichtbrechung. Grundstein im Umgang mit der Lichtbrechung, ist die entwickelte Kalibrierungsmethode, die neben den Kameraparametern auch die Brechungsebenen des Aquariums erfasst. Darauf aufbauend wurde die Lichtbrechung in verschiedene Methoden der Bildverarbeitung integriert. Für das Tracking der Fische wurde eine Ray-Tracing-basierte Methode entwickelt, die Silhouetten aus zwei kalibrierten Kameraaufnahmen unter Berücksichtigung der Lichtbrechung in einfache 3D-Objekte in Echtzeit überführen kann. Zudem wurde eine Analyse-durch-Synthese Methode entwickelt, die auf Basis der Kalibrierung des realen Systems die Lichtbrechung synthetisiert und eine genaue Rekonstruktion der Modellposition, -pose und biegung ermöglicht. Mit Hilfe dieses Verfahrens können Fischbewegungen aus Videosequenzen automatisch auf die 3D-Animationsmodelle übertragen und zur Präsentation genutzt werden.
Darüber hinaus wurde ein Methode entwickelt, die trotz welliger Wasseroberfläche und dadurch verursachter ungeordneter Lichtbrechung, ein Positionstracking von Fischen ermöglicht.
Weitere in dieser Verfahrenskette entwickelte Methoden unterstützen den Benutzer bei der Durchführung der Experimente, ermöglichen einen Austausch von Animationen und Modellen mit anderen Forschungsgruppen und erhöhen die Standardisierung.
Zur einfachen und modularen Verknüpfung aller für das System benötigten Komponenten wurde in dieser Arbeit ein aus der Robotik stammendes Framework auf die hier behandelte Thematik konzeptionell transferiert und praktisch angewendet. Dadurch bleibt das System auch für zukünftige Forschungsfragen durch einfache, modulare Erweiterbarkeit anwendbar.

6.1 Ausblick

Aufbauend auf der hier vorgestellten Arbeit lassen sich weitere Anwendungen für die Fischverhaltensforschung entwickeln. Im Bereich der interaktiven Animation eröffnet das Analyse-durch-Synthese Verfahren neue Möglichkeiten. Die aus den Bildern extrahierten Posen bilden, über die Zeit betrachtet, Ver-

haltensmuster der Fische ab. Eine Klassifikation dieser Bewegungssequenzen kann zur automatischen Aktionserkennung genutzt werden.
In diesem Kontext bietet sich eine Untersuchung an, ob aktuelle, für die menschliche Bewegung entwickelte, DeepLearning-basierte Verfahren zur Aktionserkennung wie in [125] mit Convolutional Neural Networks (CNN) oder in [158] mit Recurrent Neural Network (RNN) durchgeführt, auf Fischbewegungen zu übertragen sind und ob sich durch deren Anwendung Vorteile ergeben. Da diese Verfahren oft einen großen Umfang an annotierten Daten zum Anlernen benötigen, bildet das hier entwickelte System eine gute Grundlage, da große Mengen an Trainingsdaten von der entwickelten Simulationsumgebung synthetisiert werden können.
Eine automatische Aktionserkennung in Echtzeit würde wiederum ein mächtiges Werkzeug für die stimulusbasierte Verhaltensforschung bilden. Der virtuelle Stimulus könnte auf die erkannten Verhaltensmuster mit zuvor definierten, der Forschungsfrage entsprechenden, Mustern reagieren.
Das entwickelte Trackingsystem bietet gute Voraussetzungen, um die entwickelte Simulationsumgebung durch eine blickwinkelabhängige Simulation zu erweitern und somit den Testfischen eine noch realistischere virtuelle Welt zu generieren. Besonders das modulare, auf ROS basierte Konzept der *FishSim Animation Toolchain* bietet beste Voraussetzungen zur schnellen und einfachen Erweiterung. Hierzu trägt auch der frei verfügbare Quellcode bei, der durch interessierte Forscher kostenlos heruntergeladen und erweitert werden kann.

Literaturverzeichnis

[1] Adams, N. S. ; Rondorf, D. W. ; Evans, S. D. ; Kelly, J. E.: Effects of surgically and gastrically implanted radio transmitters on growth and feeding behavior of juvenile Chinook salmon. In: Transactions of the American Fisheries Society 127 (1998), Nr. 1, S. 128–136

[2] Ahmad, S. B. ; Gierszewski, S. ; Müller, K. ; Witte, K. ; Kuhnert, K.-D. : More action with interaction? Investigating the effect of interactive virtual 3D-fish on association time in sailfin molly, Poecilia latipinna. In: ETHO 2017, 12th Annual Meeting of the Ethologische Gesellschaft, 2017

[3] Allen, J. M. ; Nicoletto, P. F.: Response of Betta splendens to computer animations of males with fins of different length. In: Copeia 1997 (1997), Nr. 1, S. 195–199

[4] AlZu'bi, H. ; Al-Nuaimy, W. ; Buckley, J. ; Sneddon, L. ; Young, I. : Real-time 3D fish tracking and behaviour analysis. In: Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT) IEEE, 2015, S. 1–5

[5] Bakker, T. C. ; Künzler, R. : Computer animations as a tool in the study of mating preferences. In: Behaviour 135 (1998), Nr. 8, S. 1137–1159

[6] Bakker, T. C. ; Künzler, R. ; Mazzi, D. : Sexual selection: condition-related mate choice in sticklebacks. In: Nature 401 (1999), Nr. 6750, S. 234

[7] Baldauf, S. ; Kullmann, H. ; Thünken, T. ; Winter, S. ; Bakker, T. : Computer animation as a tool to study preferences in the cichlid

Pelvicachromis taeniatus. In: Journal of fish biology 75 (2009), Nr. 3, S. 738–746

[8] Baube, C. ; Rowland, W. ; Fowler, J. : The mechanisms of colour-based mate choice in female threespine sticklebacks: hue, contrast and configurational cues. In: Behaviour (1995), S. 979–996

[9] Benezeth, Y. ; Jodoin, P.-M. ; Emile, B. ; Laurent, H. ; Rosenberger, C. : Comparative study of background subtraction algorithms. In: Journal of Electronic Imaging 19 (2010), Nr. 3, S. 033003

[10] Benson, K. E.: Enhanced female brood patch size stimulates male courtship in Xiphophorus helleri. In: Copeia 2007 (2007), Nr. 1, S. 212–217

[11] Biobserve GmbH: Behavioral Research VIEWER - Behavioral Research. http://www.biobserve.com/behavioralresearch/products/viewer/, 03 2019. – abgerufen am 20.03.2019

[12] Bisazza, A. ; Piffer, L. ; Serena, G. ; Agrillo, C. : Ontogeny of numerical abilities in fish. In: PLoS One 5 (2010), Nr. 11, S. e15516

[13] Bouwmans, T. ; Porikli, F. ; Höferlin, B. ; Vacavant, A. : Background modeling and foreground detection for video surveillance. CRC press, 2014

[14] Bradski, G. R.: Real time face and object tracking as a component of a perceptual user interface. In: Proceedings Fourth IEEE Workshop on Applications of Computer Vision. WACV'98 (Cat. No. 98EX201) IEEE, 1998, S. 214–219

[15] Broida, T. J. ; Chellappa, R. : Estimation of object motion parameters from noisy images. In: IEEE Transactions on Pattern Analysis & Machine Intelligence (1986), Nr. 1, S. 90–99

[16] Brown, R. S. ; Geist, D. R. ; Deters, K. A. ; Grassell, A. : Effects of surgically implanted acoustic transmitters> 2% of body mass on the swimming performance, survival and growth of juvenile sockeye

and Chinook salmon. In: Journal of Fish Biology 69 (2006), Nr. 6, S. 1626–1638

[17] Buchanan, J. W. ; Sousa, M. C.: The edge buffer: A data structure for easy silhouette rendering. In: Proceedings of the 1st international symposium on Non-photorealistic animation and rendering ACM, 2000, S. 39–42

[18] Butail, S. ; Bartolini, T. ; Porfiri, M. : Collective response of zebrafish shoals to a free-swimming robotic fish. In: PLoS One 8 (2013), Nr. 10, S. e76123

[19] Butail, S. ; Chicoli, A. ; Paley, D. A.: Putting the fish in the fish tank: Immersive VR for animal behavior experiments. In: International Conference on Robotics and Automation IEEE, 2012, S. 5018–5023

[20] Butail, S. ; Paley, D. A.: 3D reconstruction of fish schooling kinematics from underwater video. In: International Conference on Robotics and Automation (ICRA) IEEE, 2010, S. 2438–2443

[21] Butail, S. ; Paley, D. A.: Three-dimensional reconstruction of the fast-start swimming kinematics of densely schooling fish. In: Journal of the Royal Society Interface 9 (2011), Nr. 66, S. 77–88

[22] Butkowski, T. ; Yan, W. ; Gray, A. M. ; Cui, R. ; Verzijden, M. N. ; Rosenthal, G. G.: Automated interactive video playback for studies of animal communication. In: Journal of visualized experiments: JoVE (2011), Nr. 48

[23] Cachat, J. ; Stewart, A. ; Grossman, L. ; Gaikwad, S. ; Kadri, F. ; Chung, K. M. ; Wu, N. ; Wong, K. ; Roy, S. ; Suciu, C. u. a.: Measuring behavioral and endocrine responses to novelty stress in adult zebrafish. In: Nature protocols 5 (2010), Nr. 11, S. 1786

[24] Cachat, J. ; Stewart, A. ; Utterback, E. ; Hart, P. ; Gaikwad, S. ; Wong, K. ; Kyzar, E. ; Wu, N. ; Kalueff, A. V.: Three-dimensional neurophenotyping of adult zebrafish behavior. In: PloS one 6 (2011), Nr. 3, S. e17597

[25] Cao, Z. ; Simon, T. ; Wei, S.-E. ; Sheikh, Y. : Realtime multi-person 2d pose estimation using part affinity fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, S. 7291–7299

[26] Chen, C. ; Yang, Y. ; Nie, F. ; Odobez, J.-M. : 3D human pose recovery from image by efficient visual feature selection. In: Computer Vision and Image Understanding 115 (2011), Nr. 3, S. 290–299

[27] Chen, C. ; Zhuang, Y. ; Xiao, J. ; Wu, F. : Adaptive and compact shape descriptor by progressive feature combination and selection with boosting. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on IEEE, 2008, S. 1–8

[28] Cheng, Y. : Mean shift, mode seeking, and clustering. In: IEEE transactions on pattern analysis and machine intelligence 17 (1995), Nr. 8, S. 790–799

[29] Choi, C. ; Christensen, H. u. a.: Real-time 3D model-based tracking using edge and keypoint features for robotic manipulation. In: International Conference on Robotics and Automation (ICRA) IEEE, 2010, S. 4048–4055

[30] Chouinard-Thuly, L. ; Gierszewski, S. ; Rosenthal, G. G. ; Reader, S. M. ; Rieucau, G. ; Woo, K. L. ; Gerlai, R. ; Tedore, C. ; Ingley, S. J. ; Stowers, J. R. u. a.: Technical and conceptual considerations for using animated stimuli in studies of animal behavior. In: Current Zoology 63 (2017), Nr. 1, S. 5–19

[31] Chuang, M.-C. ; Hwang, J.-N. ; Williams, K. ; Towler, R. : Tracking live fish from low-contrast and low-frame-rate stereo videos. In: IEEE Transactions on Circuits and Systems for Video Technology 25 (2015), Nr. 1, S. 167–179

[32] Cireşan, D. C. ; Giusti, A. ; Gambardella, L. M. ; Schmidhuber, J. : Mitosis detection in breast cancer histology images with deep neural networks. In: International Conference on Medical Image Computing and Computer-assisted Intervention Springer, 2013, S. 411–418

[33] Collet, A. ; Martinez, M. ; Srinivasa, S. S.: The MOPED framework: Object recognition and pose estimation for manipulation. In: The International Journal of Robotics Research (2011), S. 0278364911401765

[34] Collins, R. T. ; Lipton, A. J. ; Kanade, T. ; Fujiyoshi, H. ; Duggins, D. ; Tsin, Y. ; Tolliver, D. ; Enomoto, N. ; Hasegawa, O. ; Burt, P. u. a.: A system for video surveillance and monitoring. In: VSAM final report (2000), S. 1–68

[35] Culumber, Z. W. ; Rosenthal, G. G.: Mating preferences do not maintain the tailspot polymorphism in the platyfish, Xiphophorus variatus. In: Behavioral ecology 24 (2013), Nr. 6, S. 1286–1291

[36] Delcourt, J. ; Denoel, M. ; Ylieff, M. ; Poncin, P. : Video multitracking of fish behaviour: a synthesis and future perspectives. In: Fish and Fisheries 14 (2012), S. 186–204

[37] Deng, Z. ; Carlson, T. J. ; Li, H. ; Xiao, J. ; Myjak, M. J. ; Lu, J. ; Martinez, J. J. ; Woodley, C. M. ; Weiland, M. A. ; Eppard, M. B.: An injectable acoustic transmitter for juvenile salmon. In: Scientific reports 5 (2015), S. 8111

[38] Dennis Jr, J. E. ; Gay, D. M. ; Walsh, R. E.: An adaptive nonlinear least-squares algorithm. In: ACM Transactions on Mathematical Software (TOMS) 7 (1981), Nr. 3, S. 348–368

[39] Denoël, M. ; Bichot, M. ; Ficetola, G. F. ; Delcourt, J. ; Ylieff, M. ; Kestemont, P. ; Poncin, P. : Cumulative effects of road de-icing salt on amphibian behavior. In: Aquatic Toxicology 99 (2010), Nr. 2, S. 275–280

[40] Dollar, P. ; Wojek, C. ; Schiele, B. ; Perona, P. : Pedestrian detection: An evaluation of the state of the art. In: IEEE transactions on pattern analysis and machine intelligence 34 (2012), Nr. 4, S. 743–761

[41] Douglas, D. H. ; Peucker, T. K.: Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. In: Cartographica: the international journal for geographic information and geovisualization 10 (1973), Nr. 2, S. 112–122

[42] Eberly, D. H.: 3D game engine design: a practical approach to real-time computer graphics. San Francisco u.a. : Kaufmann, 2001

[43] Faria, J. J. ; Dyer, J. R. ; Clément, R. O. ; Couzin, I. D. ; Holt, N. ; Ward, A. J. ; Waters, D. ; Krause, J. : A novel method for investigating the collective behaviour of fish: introducing ?Robofish? In: Behavioral Ecology and Sociobiology 64 (2010), Nr. 8, S. 1211–1218

[44] Fischer, S. ; Taborsky, B. ; Burlaud, R. ; Fernandez, A. A. ; Hess, S. ; Oberhummer, E. ; Frommen, J. G.: Animated images as a tool to study visual communication: a case study in a cooperatively breeding cichlid. In: Behaviour 151 (2014), Nr. 12-13, S. 1921–1942

[45] Fisher, R. B. ; Chen-Burger, Y.-H. ; Giordano, D. ; Hardman, L. ; Lin, F.-P. u. a.: Fish4Knowledge: collecting and analyzing massive coral reef fish video data. Bd. 104. Springer, 2016

[46] Fleishman, L. ; Endler, J. : Some comments on visual perception and the use of video playback in animal behavior studies. In: Acta ethologica 3 (2000), Nr. 1, S. 15–27

[47] Gao, X.-S. ; Hou, X.-R. ; Tang, J. ; Cheng, H.-F. : Complete solution classification for the perspective-three-point problem. In: IEEE transactions on pattern analysis and machine intelligence 25 (2003), Nr. 8, S. 930–943

[48] Gierszewski, S. ; Baker, D. ; Müller, K. ; Hütwohl, J.-M. ; Kuhnert, K.-D. ; Witte, K. : Using the FishSim Animation Toolchain to Investigate Fish Behavior: A Case Study on Mate-Choice Copying In Sailfin Mollies. In: Journal of Visualized Experiments: JoVE (2018), Nr. 141. http://dx.doi.org/10.3791/58435. – DOI 10.3791/58435

[49] Gierszewski, S. ; Müller, K. ; Smielik, I. ; Hütwohl, J.-M. ; Kuhnert, K.-D. ; Witte, K. : The virtual lover: variable and easily guided 3D fish animations as an innovative tool in mate-choice experiments with sailfin mollies-II. Validation. In: Current Zoology 63 (2017), Nr. 1, S. 65–74. http://dx.doi.org/10.1093/cz/zow108. – DOI 10.1093/cz/zow108

[50] Henrion, S. ; Spoor, C. W. ; Pieters, R. P. ; Müller, U. K. ; Leeuwen, J. L.: Refraction corrected calibration for aquatic locomotion research: application of Snell's law improves spatial accuracy. In: Bioinspiration & biomimetics 10 (2015), Nr. 4, S. 046009

[51] Hinterstoisser, S. ; Benhimane, S. ; Navab, N. : N3m: Natural 3d markers for real-time object detection and pose estimation. In: 11th International Conference on Computer Vision IEEE, 2007, S. 1–7

[52] Ho, C. ; Shih, C. : Real-time tracking and stereo vision-based control of a goldfish-catching system. In: Proceedings of the 35th International MATADOR Conference Springer, 2007, S. 319–322

[53] Horn, B. K. ; Schunck, B. G.: Determining optical flow. In: Artificial intelligence 17 (1981), Nr. 1-3, S. 185–203

[54] Ingley, S. J. ; Asl, M. R. ; Wu, C. ; Cui, R. ; Gadelhak, M. ; Li, W. ; Zhang, J. ; Simpson, J. ; Hash, C. ; Butkowski, T. u. a.: anyFish 2.0: an open-source software platform to generate and share animated fish models to study behavior. In: SoftwareX 3 (2015), S. 13–21

[55] Jakka, N. ; Gnaneshwar Rao, T. ; Venkateswara Rao, J. : Locomotor behavioral response of mosquitofish (Gambusia affinis) to subacute mercury stress monitored by video tracking system. In: Drug and chemical toxicology 30 (2007), Nr. 4, S. 383–397

[56] Kalman, R. E.: A new approach to linear filtering and prediction problems. In: Journal of basic Engineering 82 (1960), Nr. 1, S. 35–45

[57] Kawahara, R. ; Nobuhara, S. ; Matsuyama, T. : Dynamic 3D capture of swimming fish by underwater active stereo. In: Methods in Oceanography 17 (2016), S. 118–137

[58] Kim, H. ; Sakamoto, R. ; Kitahara, I. ; Toriyama, T. ; Kogure, K. : Robust foreground extraction technique using Gaussian family model and multiple thresholds. In: Asian Conference on Computer Vision Springer, 2007, S. 758–768

[59] Kim, K. ; Chalidabhongse, T. H. ; Harwood, D. ; Davis, L. : Background modeling and subtraction by codebook construction. In: Image Processing, 2004. ICIP'04. 2004 International Conference on Bd. 5 IEEE, 2004, S. 3061–3064

[60] Kim, K. ; Chalidabhongse, T. H. ; Harwood, D. ; Davis, L. : Real-time foreground–background segmentation using codebook model. In: Real-time imaging 11 (2005), Nr. 3, S. 172–185

[61] Klette, R. ; SchlÃ$\frac{1}{4}$ns, K. ; Koschan, A. : Computer vision - three dimensional data from images. Berlin: Springer, 1998

[62] Kulkarni, T. D. ; Kohli, P. ; Tenenbaum, J. B. ; Mansinghka, V. : Picture: A probabilistic programming language for scene perception. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, S. 4390–4399

[63] Kunze, J. ; Hartmann, S. ; Witte, K. ; Kuhnert, K.-D. : Daphnia magna as biosensor for Ag-nanoparticles in water systems. In: Fisher, B. (Hrsg.) ; Cancun, Mexico (Veranst.): 23rd International Conference on Pattern Recognition; VAIB 2016 - Visual observation and analysis of Vertebrate and Insect Behavior Workshop Proceedings Bd. 23 Cancun, Mexico, 2016, S. 5

[64] Künzler, R. ; Bakker, T. C.: Female preferences for single and combined traits in computer animated stickleback males. In: Behavioral Ecology 12 (2001), Nr. 6, S. 681–685

[65] Landgraf, T. ; Bierbach, D. ; Nguyen, H. ; Muggelberg, N. ; Romanczuk, P. ; Krause, J. : RoboFish: increased acceptance of interactive robotic fish with realistic eyes and natural motion patterns by live Trinidadian guppies. In: Bioinspiration & biomimetics 11 (2016), Nr. 1, S. 015001

[66] Landgraf, T. ; Nguyen, H. ; Schröer, J. ; Szengel, A. ; Clément, R. J. ; Bierbach, D. ; Krause, J. : Blending in with the shoal: robotic fish swarms for investigating strategies of group formation in guppies.

In: Conference on Biomimetic and Biohybrid Systems Springer, 2014, S. 178–189

[67] Laurel, B. J. ; Laurel, C. J. ; Brown, J. A. ; Gregory, R. S.: A new technique to gather 3D spatial information using a single camera. In: Journal of Fish Biology 66 (2005), S. 429–441

[68] Liu, C. ; Yuen, J. ; Torralba, A. : Sift flow: Dense correspondence across scenes and its applications. In: IEEE transactions on pattern analysis and machine intelligence 33 (2011), Nr. 5, S. 978–994

[69] Lucas, B. D. ; Kanade, T. : Optical navigation by the method of differences. In: Proceedings of the 9th international joint conference on Artificial intelligence-Volume 2 Morgan Kaufmann Publishers Inc., 1985, S. 981–984

[70] Maggio, E. ; Cavallaro, A. : Video tracking: theory and practice. John Wiley & Sons, 2011

[71] Maggio, E. ; Cavallaro, A. : What is video tracking? In: Video Tracking: Theory and Practice (2011), S. 1–14

[72] Mazzi, D. ; Künzler, R. ; Bakker, T. C.: Female preference for symmetry in computer-animated three-spined sticklebacks, Gasterosteus aculeatus. In: Behavioral Ecology and Sociobiology 54 (2003), Nr. 2, S. 156–161

[73] Mazzi, D. ; Künzler, R. ; Largiadèr, C. R. ; Bakker, T. C.: Inbreeding affects female preference for symmetry in computer-animated sticklebacks. In: Behavior Genetics 34 (2004), Nr. 4, S. 417–424

[74] McDonald, C. ; Reimchen, T. ; Hawryshyn, C. : Nuptial colour loss and signal masking in Gasterosteus: an analysis using video imaging. In: Behaviour 132 (1995), Nr. 13, S. 963–978

[75] McKinnon, J. S.: Video mate preferences of female three-spined sticklebacks from populations with divergent male coloration. In: Animal Behaviour 50 (1995), Nr. 6, S. 1645–1655

[76] McKinnon, J. S. ; McPhail, J. : Male aggression and colour in divergent populations of the threespine stickleback: experiments with animations. In: Canadian journal of zoology 74 (1996), Nr. 9, S. 1727–1733

[77] Mehlis, M. ; Bakker, T. C. ; Frommen, J. G.: Smells like sib spirit: kin recognition in three-spined sticklebacks (Gasterosteus aculeatus) is mediated by olfactory cues. In: Animal Cognition 11 (2008), Nr. 4, S. 643–650

[78] Möller, T. ; Trumbore, B. : Fast, minimum storage ray/triangle intersection. In: ACM SIGGRAPH 2005 Courses ACM, 2005, S. 7

[79] Morais, E. F. ; Campos, M. F. M. ; Padua, F. L. ; Carceroni, R. L.: Particle filter-based predictive tracking for robust fish counting. In: XVIII Brazilian Symposium on Computer Graphics and Image Processing (SIBGRAPI'05) IEEE, 2005, S. 367–374

[80] Moré, J. J.: The Levenberg-Marquardt algorithm: implementation and theory. In: Numerical analysis. Springer, 1978, S. 105–116

[81] Morris, M. R. ; Moretz, J. A. ; Farley, K. ; Nicoletto, P. : The role of sexual selection in the loss of sexually selected traits in the swordtail fish Xiphophorus continens. In: Animal Behaviour 69 (2005), Nr. 6, S. 1415–1424

[82] Morris, M. R. ; Nicoletto, P. F. ; Hesselman, E. : A polymorphism in female preference for a polymorphic male trait in the swordtail fish Xiphophorus cortezi. In: Animal Behaviour 65 (2003), Nr. 1, S. 45–52

[83] Müller, K. ; Akupati, C. R. ; Graf, F. ; Guo, Y. ; Höhn, S. ; Hütwohl, J.-M. ; Köther, T. ; Osman, S. N. ; Poenaru, G. ; Sedighi, S. ; Schlemper, J.-F. ; Zhu, W. ; Kunze, J. ; Kuhnert, K.-D. : Zephyr - University of Siegen, Germany. In: Rakun, J. (Hrsg.): Proceedings of the 13th Field Robot Event 2015, University of Maribor : Faculty of Agriculture and Life Sciences, Department of Biosystems Engineering, 2015, 122-132

[84] Müller, K. ; Behrozin, R. ; Höhn, S. ; Hütwohl, J. M. ; Köther, T. ; Rothenpieler, T. ; Schmidt, F. ; Wesener, T. ; Zhu, W. ;

Kuhnert, K.-D. : Team Phaethon - University of Siegen. In: Arellano, M. V. (Hrsg.) ; Griepentrog, H. W. (Hrsg.): Proceedings of the 12th Field Robot Event 2014 Bd. 12, University of Hohenheim, Instrumentation & Test Engineering (440c) Stuttgart, Germany, 4 2015 (12 1), S. 141–151

[85] Müller, K. ; Gierszewski, S. ; Witte, K. ; Kuhnert, K.-D. : Where is my mate? Real-time 3-D fish tracking for interactive mate-choice experiments. In: 23rd International Conference on Pattern Recognition; VAIB 2016 - Visual observation and analysis of Vertebrate and Insect Behavior Workshop Proceedings, 2016

[86] Müller, K. ; Hütwohl, J.-M. ; Gierszewski, S. ; Witte, K. ; Kuhnert, K.-D. : Fish Motion Capture with Refraction Synthesis. In: Proceedings of 26. International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision WSCG Bd. 26, 2018, S. 125–134

[87] Müller, K. ; Schlemper, J. ; Kuhnert, L. ; Kuhnert, K.-D. : Calibration and 3D ground truth data generation with orthogonal camera-setup and refraction compensation for aquaria in real-time. In: International Conference on Computer Vision Theory and Applications (VISAPP) Bd. 3 IEEE, 2014, S. 626–634

[88] Müller, K. ; Smielik, I. ; Hütwohl, J.-M. ; Gierszewski, S. ; Witte, K. ; Kuhnert, K.-D. : The virtual lover: variable and easily guided 3D fish animations as an innovative tool in mate-choice experiments with sailfin mollies-I. Design and implementation. In: Current Zoology 63 (2016), Nr. 1, S. 55–64. http://dx.doi.org/10.1093/cz/zow106. – DOI 10.1093/cz/zow106

[89] Müller, K. ; Smielik, I. ; Kuhnert, K.-D. : Optimal Feature-set Selection Controlled by Pose-space Location. In: International Conference on Computer Vision Theory and Applications (VISAPP) IEEE, 2016, S. 200–207

[90] Nakayasu, T. ; Watanabe, E. : Biological motion stimuli are attractive to medaka fish. In: Animal cognition 17 (2014), Nr. 3, S. 559–575

[91] Nguyen, D. ; Kuhnert, L. ; Jiang, T. ; Thamke, S. ; Kuhnert, K. : Vegetation detection for outdoor automobile guidance. In: International Conference on Industrial Technology (ICIT) IEEE, 2011, S. 358–364

[92] Noldus, L. P. ; Spink, A. J. ; Tegelenbosch, R. A.: EthoVision: a versatile video tracking system for automation of behavioral experiments. In: Behavior Research Methods, Instruments, & Computers 33 (2001), Nr. 3, S. 398–414

[93] Noldus Information Technology: New in EthoVision XT 14 | Noldus. https://www.noldus.com/EthoVision-XT/New, 01 2019. – abgerufen am 07.03.2019

[94] Noldus Information Technology: Track 3D. https://www.noldus.com/animal-behavior-research/products/track3d, 3 2019. – abgerufen am 13.03.2019

[95] Oliveira, R. F. ; Rosenthal, G. G. ; Schlupp, I. ; McGregor, P. K. ; Cuthill, I. C. ; Endler, J. A. ; Fleishman, L. J. ; Zeil, J. ; Barata, E. ; Burford, F. u. a.: Considerations on the use of video playbacks as visual stimuli: the Lisbon workshop consensus. In: Acta ethologica 3 (2000), Nr. 1, S. 61–65

[96] OpenCV team: OpenCV library. https://opencv.org/, 03 2019. – abgerufen am 29.03.2019

[97] Palmér, T. ; Aström, K. ; Enqvist, O. ; Petersson, P. : Visual Analysis of Zebrafish Behavior. In: Conference on Visual observation and analysis of animal and insect behavior Bd. 1, 2016

[98] Payet, N. ; Todorovic, S. : From contours to 3d object detection and pose estimation. In: International Conference on Computer Vision (ICCV) IEEE, 2011, S. 983–990

[99] Pedersen, M. ; Hein Bengtson, S. ; Gade, R. ; Madsen, N. ; Moeslund, T. B.: Camera Calibration for Underwater 3D Reconstruction Based on Ray Tracing Using Snell's Law. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2018, S. 1410–1417

[100] Pinhasov, A. ; Crooke, J. ; Rosenthal, D. ; Brenneman, D. ; Malatynska, E. : Reduction of Submissive Behavior Model for antidepressant drug activity testing: study using a video-tracking system. In: Behavioural pharmacology 16 (2005), Nr. 8, S. 657–664

[101] Pinkiewicz, T. ; Purser, G. ; Williams, R. : A computer vision system to analyse the swimming behaviour of farmed fish in commercial aquaculture facilities: a case study using cage-held Atlantic salmon. In: Aquacultural Engineering 45 (2011), Nr. 1, S. 20–27

[102] Pons-Moll, G. ; Baak, A. ; Helten, T. ; Müller, M. ; Seidel, H.-P. ; Rosenhahn, B. : Multisensor-fusion for 3d full-body human motion capture. In: Conference onComputer Vision and Pattern Recognition (CVPR) IEEE, 2010, S. 663–670

[103] Poppe, R. : Vision-based human motion analysis: An overview. In: Computer vision and image understanding 108 (2007), Nr. 1-2, S. 4–18

[104] Poppe, R. ; Poel, M. : Comparison of silhouette shape descriptors for example-based human pose recovery. In: Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on IEEE, 2006, S. 541–546

[105] Quigley, M. ; Conley, K. ; Gerkey, B. ; Faust, J. ; Foote, T. ; Leibs, J. ; Wheeler, R. ; Ng, A. Y.: ROS: an open-source Robot Operating System. In: ICRA workshop on open source software Bd. 3 Kobe, Japan, 2009, S. 5

[106] Rasines, I. ; Remazeilles, A. ; Iriondo Bengoa, P. M.: Feature selection for hand pose recognition in human-robot object exchange scenario. In: Emerging Technology and Factory Automation (ETFA), 2014 IEEE IEEE, 2014, S. 1–8

[107] Reinbacher, C. ; Ruther, M. ; Bischof, H. : Pose estimation of known objects by efficient silhouette matching. In: 20th International Conference on Pattern Recognition IEEE, 2010, S. 1080–1083

[108] Richmond, J. u. a.: The three Rs. In: The UFAW handbook on the care and management of laboratory and other research animals (2010), S. 5–22

[109] Roetenberg, D. ; Luinge, H. ; Slycke, P. : Xsens MVN: full 6DOF human motion tracking using miniature inertial sensors. In: Xsens Motion Technologies BV, Tech. Rep (2009)

[110] ROS: Is ROS For Me? http://www.ros.org/is-ros-for-me/, 2018. – abgerufen am 08.12.2018

[111] Rosenthal, G. G.: Design considerations and techniques for constructing video stimuli. In: Acta ethologica 3 (2000), Nr. 1, S. 49–54

[112] Rosenthal, G. G. ; Evans, C. S. ; Miller, W. L.: Female preference for dynamic traits in the green swordtail, Xiphophorus helleri. In: Animal Behaviour 51 (1996), Nr. 4, S. 811–820

[113] Rowland, W. J. ; Bolyard, K. J. ; Jenkins, J. J. ; Fowler, J. : Video playback experiments on stickleback mate choice: female motivation and attentiveness to male colour cues. In: Animal Behaviour 49 (1995), Nr. 6, S. 1559–1567

[114] Saberioon, M. ; Cisar, P. : Automated multiple fish tracking in three-dimension using a structured light sensor. In: Computers and Electronics in Agriculture 121 (2016), S. 215–221

[115] Salierno, J. ; Gipson, G. ; Kane, A. : Quantitative movement analysis of social behavior in mummichog, Fundulus heteroclitus. In: Journal of Ethology 26 (2008), Nr. 1, S. 35–42

[116] Sanin, A. ; Sanderson, C. ; Lovell, B. C.: Shadow detection: A survey and comparative evaluation of recent methods. In: Pattern recognition 45 (2012), Nr. 4, S. 1684–1695

[117] Schlupp, I. ; Marler, C. ; Ryan, M. J.: Benefit to male sailfin mollies of mating with heterospecific females. In: Science 263 (1994), Nr. 5145, S. 373–374

[118] Schlupp, I. ; Ryan, M. J.: Male sailfin mollies (Poecilia latipinna) copy the mate choice of other males. In: Behavioral Ecology 8 (1997), Nr. 1, S. 104–107

[119] Schweitzer, H. ; Bell, J. W. ; Wu, F. : Very fast template matching. In: European Conference on Computer Vision Springer, 2002, S. 358–372

[120] Shaikh, S. H. ; Chaki, N. ; Saeed, K. : Moving Object Detection Using Background Subtraction. Cham : Springer, 2014

[121] Shannon, C. E.: Communication in the presence of noise. In: Proceedings of the IRE 37 (1949), Nr. 1, S. 10–21

[122] Shashar, N. ; Rutledge, P. ; Cronin, T. : Polarization vision in cuttlefish in a concealed communication channel? In: Journal of Experimental Biology 199 (1996), Nr. 9, S. 2077–2084

[123] Shotton, J. ; Sharp, T. ; Kipman, A. ; Fitzgibbon, A. ; Finocchio, M. ; Blake, A. ; Cook, M. ; Moore, R. : Real-time human pose recognition in parts from single depth images. In: Communications of the ACM 56 (2013), Nr. 1, S. 116–124

[124] Sibert, J. : Electronic tagging and tracking in marine fisheries. In: Electronic Tagging and Tracking in Marine Fisheries. Springer, 2001, S. 1–6

[125] Simonyan, K. ; Zisserman, A. : Two-stream convolutional networks for action recognition in videos. In: Advances in neural information processing systems, 2014, S. 568–576

[126] Smielik, I. ; Müller, K. ; Kuhnert, K.-D. : Fish motion simulation. In: European Simulation and Modelling Conference (ESM), 2015, S. 392–396

[127] Sminchisescu, C. ; Telea, A. : Human pose estimation from silhouettes. a consistent approach using distance level sets. In: 10th International Conference on Computer Graphics, Visualization and Computer Vision (WSCG'02) Bd. 10, 2002

[128] Smith, A. R.: Color gamut transform pairs. In: ACM Siggraph Computer Graphics 12 (1978), Nr. 3, S. 12–19

[129] Stauffer, C. ; Grimson, W. E. L.: Adaptive background mixture models for real-time tracking. In: Proceedings. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No PR00149) Bd. 2 IEEE, 1999, S. 246–252

[130] Stewart, A. M. ; Grieco, F. ; Tegelenbosch, R. A. ; Kyzar, E. J. ; Nguyen, M. ; Kaluyeva, A. ; Song, C. ; Noldus, L. P. ; Kalueff, A. V.: A novel 3D method of locomotor analysis in adult zebrafish: Implications for automated detection of CNS drug-evoked phenotypes. In: Journal of neuroscience methods 255 (2015), S. 66–74

[131] Stowers, J. R. ; Fuhrmann, A. ; Hofbauer, M. ; Streinzer, M. ; Schmid, A. ; Dickinson, M. H. ; Straw, A. D.: Reverse Engineering Animal Vision with Virtual Reality and Genetics. In: Computer 47 (2014), July, Nr. 7, S. 38–45

[132] Stowers, J. R. ; Hofbauer, M. ; Bastien, R. ; Griessner, J. ; Higgins, P. ; Farooqui, S. ; Fischer, R. M. ; Nowikovsky, K. ; Haubensak, W. ; Couzin, I. D. u. a.: Virtual reality for freely moving animals. In: Nature methods 14 (2017), Nr. 10, S. 995

[133] Süsse, H. ; Rodner, E. : Bildverarbeitung und Objekterkennung. Wiesbaden : Springer Vieweg, 2014

[134] Suzuki, K. ; Takagi, T. ; Hiraishi, T. : Video analysis of fish schooling behavior in finite space using a mathematical model. In: Fisheries Research 60 (2003), Nr. 1, S. 3–10

[135] Suzuki, S. : Topological structural analysis of digitized binary images by border following. In: Computer vision, graphics, and image processing 30 (1985), Nr. 1, S. 32–46

[136] Takahashi, H. ; Hatoya, J. ; Hashimoto, N. ; Nakajima, M. : Animation synthesis for virtual fish from video. In: Proceedings of the 10th ICAT (International Conference on Artificial reality and Telexistence), 2000, S. 90–97

[137] Tedore, C. ; Johnsen, S. : Using RGB displays to portray color realistic imagery to animal eyes. In: Current Zoology 63 (2017), Nr. 1, S. 27–34

[138] Ter Pelkwijk, J. J. ; Tinbergen, N. : Eine reizbiologische Analyse einiger Verhaltensweisen von Gasterosteus aculeatus L. In: Zeitschrift für Tierpsychologie 1 (1937), Nr. 3, S. 193–200

[139] Thoen, H. H. ; How, M. J. ; Chiou, T.-H. ; Marshall, J. : A different form of color vision in mantis shrimp. In: Science 343 (2014), Nr. 6169, S. 411–413

[140] Thünken, T. ; Baldauf, S. A. ; Kullmann, H. ; Schuld, J. ; Hesse, S. ; Bakker, T. C.: Size-related inbreeding preference and competitiveness in male Pelvicachromis taeniatus (Cichlidae). In: Behavioral Ecology 22 (2011), Nr. 2, S. 358–362

[141] Tu, X. ; Terzopoulos, D. : Artificial fishes: Physics, locomotion, perception, behavior. In: Proceedings of the 21st annual conference on Computer graphics and interactive techniques ACM, 1994, S. 43–50

[142] Veen, T. ; Ingley, S. J. ; Cui, R. ; Simpson, J. ; Asl, M. R. ; Zhang, J. ; Butkowski, T. ; Li, W. ; Hash, C. ; Johnson, J. B. u. a.: anyFish: an open-source software to generate animated fish models for behavioural studies. In: Evolutionary Ecology Research 15 (2013), Nr. 3, S. 361–375

[143] Vezhnevets, V. ; Sazonov, V. ; Andreeva, A. : A survey on pixel-based skin color detection techniques. In: Proc. Graphicon Bd. 3 Moscow, Russia, 2003, S. 85–92

[144] Viola, P. ; Jones, M. J.: Robust real-time face detection. In: International journal of computer vision 57 (2004), Nr. 2, S. 137–154

[145] Viscido, S. V. ; Parrish, J. K. ; Grünbaum, D. : Individual behavior and emergent properties of fish schools: a comparison of observation and theory. In: Marine Ecology Progress Series 273 (2004), S. 239–249

[146] Voesenek, C. J. ; Pieters, R. P. ; Leeuwen, J. L.: Automated reconstruction of three-dimensional fish motion, forces, and torques. In: PloS one 11 (2016), Nr. 1, S. e0146682

[147] Wehner, R. : Polarization vision–a uniform sensory capacity? In: Journal of Experimental Biology 204 (2001), Nr. 14, S. 2589–2596

[148] Wei, S.-E. ; Ramakrishna, V. ; Kanade, T. ; Sheikh, Y. : Convolutional pose machines. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, S. 4724–4732

[149] Witte, K. ; Ryan, M. J.: Male body length influences mate-choice copying in the sailfin molly Poecilia latipinna. In: Behavioral Ecology 9 (1998), Nr. 5, S. 534–539

[150] Wong, B. B. ; Rosenthal, G. G.: Female disdain for swords in a swordtail fish. In: The American Naturalist 167 (2005), Nr. 1, S. 136–140

[151] Yamaguchi, A. : ROS: A Minimum Overview. `http://akihikoy.net/notes/?text%2FROS%2FMinOverview`, 2018. – abgerufen am 08.11.2018

[152] Yamashita, A. ; Kawanishi, R. ; Koketsu, T. ; Kaneko, T. ; Asama, H. : Underwater sensing with omni-directional stereo camera. In: International Conference on Computer Vision Workshops (ICCV Workshops) IEEE, 2011, S. 304–311

[153] Yilmaz, A. ; Javed, O. ; Shah, M. : Object tracking: A survey. In: Acm computing surveys (CSUR) 38 (2006), Nr. 4, S. 13

[154] Zhang, Z. : A flexible new technique for camera calibration. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000), Nov, Nr. 11, S. 1330–1334

[155] Zhang, Z. : Microsoft kinect sensor and its effect. In: IEEE multimedia 19 (2012), Nr. 2, S. 4–10

[156] Zheng, W. ; Liang, L. : Fast car detection using image strip features. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on IEEE, 2009, S. 2703–2710

[157] Zhu, L. ; Weng, W. : Catadioptric stereo-vision system for the real-time monitoring of 3D behaviour in aquatic animals. In: Physiology & Behavior 91 (2007), S. 106–119

[158] Zhu, W. ; Lan, C. ; Xing, J. ; Zeng, W. ; Li, Y. ; Shen, L. ; Xie, X. : Co-occurrence feature learning for skeleton based action recognition using regularized deep LSTM networks. In: Thirtieth AAAI Conference on Artificial Intelligence, 2016

[159] Zivkovic, Z. : Improved adaptive Gaussian mixture model for background subtraction. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. Bd. 2 IEEE, 2004, S. 28–31

Glossar[1]

Bayer-Matrix Konzept, welches bei vielen Farbkameras Anwendung findet und die Verteilung der grünen, roten und blauen Sensorelemente auf einem bildgebenden Sensor definiert. Die Aufteilung der Grundfarben Rot (25 %), Grün (50 %) und Blau (25 %) ist an die des menschlichen Auges angelehnt.

Bildrauschen Zufällige Abweichung der Helligkeits- oder Farbinformation von Pixeln einer Aufnahme, welche zur Verschlechterung des Bildes führt. Bildrauschen ist zumeist durch den Sensor bedingt.

BLOB-Analyse *Binary Large Object* (BLOB) - Analyse dient der Erkennung von zusammenhängenden Bereichen im Bild. Es werden Messwerte wie Position und Fläche für einzelne BLOBs berechnet.

Boundigbox Quader (3D) oder Rechteck (2D), der/das ein Objekt vollständig umschließt.

Brute-Force-Methode Brute force - engl. rohe Gewalt. Ist eine Lösungsmethode bei der durch Ausprobieren (fast) aller möglichen Fälle eine Lösung gesucht wird.

Chrominanz Ist ein Maß aus der Videotechnik, wird auch als Farbigkeit bezeichnet und gibt Informationen über die Farbart Zusammenfassung von Farbe und Sättigung.

Deep Learning Beschreibt eine Klasse von Optimierungsmethoden, die auf neuronalen Netzen basieren und auf Basis von großen Datenmengen an-

[1]Die in diesem Glossar beinhalteten Erklärungen sollen dem fachfremden Leser beim Verständnis der Dissertation helfen und sind teils stark vereinfacht.

gelernt werden. Ein angelerntes System ist in der Lage im gelernten Bereich Prognosen abzugeben und Entscheidungen zu treffen.

Echtzeit Eigenschaft eines informationstechnischen Systems, welches Daten innerhalb einer vorgegebenen Zeitspanne zuverlässig verarbeitet und in der Prozesskette ständig betriebsbereit ist.

Entropie Beschreibt den mittleren Informationsgehalt einer Nachricht. Der Informationsgehalt wiederum gibt an, wie viel Information in einer Nachricht übertragen wird.

Epipolargeometrie Mathematisches Modell, welches die Beziehung zweier Kamerabilder unterschiedlicher Kameras desselben Objektes beschreibt. Durch die Epipolargeometrie lassen Abhängigkeiten zwischen korrespondierenden Bildpunkten eines Objektes beschreiben. Sie wird oft in Verfahren zur Gewinnung von 3D-Daten aus Bildern eingesetzt.

Ethernet Technologie, die ursprünglich für lokale Netzwerke (LANs) entworfen wurde. Sie definiert sowohl Hardware als auch Software für kabelgebundene Kommunikation in Netzwerken und wird oft als Synonym für Netzwerkverbindungen zwischen Computern genannt.

Extrusion Beschreibt den Prozess der Verwandlung einer 2D-Form in ein 3D-Volumen. Dabei wird die 2D-Form entlang einer zur Oberfläche orthogonalen Achse ”ausgezogen“.

Farbsegmentierung Beschreibt die Segmentierung eines Bereiches anhand einer Farbe. Im Bereich der Bildverarbeitung nutzt man die Farbsegmentierung, um einheitlich eingefärbte Bereiche eines Bildes in einem Segment zu vereinen.

Frame Als Frame wird ein einzelnes Bild aus einer Videosequenz bezeichnet.

Framework Siehe *Software-Framework*.

Freistellen Bezeichnet in der Bildbearbeitung die Trennung eines Objektes oder eines Bereiches vom restlichen Bild. Oft werden die ausgewählten Bereiche separat weiterverarbeitet.

Game-Engine →Software-Framework für Computerspiele, welches sowohl die visuelle Darstellung als auch den Ablauf eines Spiels steuert. Oft verfügen Game-Engines auch über Module zur Simulation physikalischer Prozesse und Entwicklungsumgebungen zur Programmierung der Spiele.

Gamecontroller Spezielles Eingabegerät zur Steuerung von Computerspielen. Oft verfügen Gamecontroller über speziell für die Spielbedienung abgestimmte Knöpfe und Hebel, die ergonomisch sinnvoll angeordnet sind.

Gauß-Filter Frequenzfilter, der unter anderem in der digitalen Signalverarbeitung und in der Bildverarbeitung eingesetzt wird.

Generisches Stimulusmodell Ist ein 3D-Computermodell eines Stimulus, welches allgemein gehalten ist und durch verschiedene Parameter für den speziellen Anwendungsfall angepasst werden kann.

Gonopodium Extremität, welche zur Begattung dient. Im Falle der Knochenfische handelt es sich um einen umgewandelten Strahl der Afterflosse.

Grafische Benutzeroberfläche engl. *graphical user interface (GUI)*. Benutzerschnittstelle zur Steuerung von Anwendungssoftware. Zur Bedienung werden grafische Symbole und Steuerelemente genutzt. Als Steuergerät kommen bei klassischen Computern Mäuse zum Einsatz. Auf Smartphones und Tablets findet die Steuerung durch Berührung statt.

HSV Farbmodell bei dem Farben mit Hilfe von drei Parametern definiert werden: Farbwert (**H**ue), Farbsättigung (**S**aturation) und Hellwert (**V**alue). Das HSV-Modell ähnelt eher der menschlichen Farbwahrnehmung als z.B. das RGB-Modell. Weitere Informationen zu diesem Farbmodell sind auch unter [128] finden.

Kamerakalibrierung Bei der Kamerakalibrierung werden die Kameraparameter ermittelt, mit denen die Abbildung der Szene auf die Bildebene (*intrinsische Kalibrierung*) und die Position und Ausrichtung der Kamera in der Welt (**extrinsische Kalibrierung**) beschrieben wird.

Key-Frame-Animation Eine Methode zur Animation von Objekten oder Modellen. Der Benutzer definiert Objektparameter wie Position oder Ausrichtung zu einem bestimmten Zeitpunkt und der Computer berechnet und generiert die Frames zwischen diesen "Key-Frames". Diese Methode reduziert den Aufwand des Animators, da er nicht jeden Frame einer Animation einzeln, sondern nur Frames an signifikante Positionen definieren kann.

Kopieren der Partnerwahl engl. *Mate Choice Copying (MCC)*. Beschreibt eine nicht unabhängige Entscheidung bei der Partnerwahl. Diese tritt auf, wenn sich die Wahrscheinlichkeit einer Paarung zwischen einem beobachtenden Individuum und einem Zielpartner positiv oder negativ aufgrund einer beobachteten Paarung zwischen dem Zielpartner und einem anderen Individuum verändert.

Käfig-Transformation Spezielles Bearbeitungswerkzeug in *GIMP*. Es ermöglicht dem Benutzer einen ausgewählten Bereich eines Bildes durch verschieben der Außengrenzen, zu verformen.

Luminanz Maß, welches im Bereich der Videotechnik die Helligkeit eines Pixels angibt.

Maske (Bildverarbeitung) Dient der Maskierung eines bestimmten Bildbereiches (der Teilmenge eines Bildes). Die maskierten Bildbereiche werden oft separat vom restlichen Bild bearbeitet.

Merkmalspunkt (Bildverarbeitung) Beschreibt einen markanten Punkt im Bild, der durch eine spezielle Metrik beschrieben und auf Grund dieser auch in anderen Bildern (insoweit diese den Merkmalspunkt zeigen) automatisch wiedergefunden werden kann.

Mesh (Polygon) Ein Polygonmesh stellt die Oberfläche eines 3D-Objekts dar. Es besteht aus Kanten und Eckpunkten, die mit einer Art Netz verbunden sind.

Motion-Capture Verfahren zur Analyse und Aufzeichnung der Bewegung von Objekten, Tieren oder Menschen.

Optischer Fluss Siehe Seite 42ff.

Pixel Ist ein Bildpunkt eines digitalen Bildes (Rastergrafik). Es beinhaltet Intensitäts- oder Farbinformationen.

Polynom Beschreibt einen mathematischen Ausdruck, der aus Variablen und Koeffizienten besteht, die nur Operationen wie Addition, Subtraktion, Multiplikation und positive Exponenten der Variablen beinhaltet. Beispiele für Polynome sind $2x + 4$ oder $4x^2 + 2x + 1$. Eine Funktion, die durch Berechnung eines Polynoms definiert ist wird auch Polynomfunktion genannt.

Pose (Robotik) Beschreibt die Kombination aus Position und Lage eines Objektes.

Proportionalregler Kurz P-Regler. Beschreibt ein Reglungssystem, dessen Ausgangsgröße proportional von der Regeldifferenz (Differenz zwischen Soll- und Istwert) abhängt.

QR-Code engl. ***Q****uick* ***R****esponse*-Code. 2D-Code, der aufgrund einer automatischen Fehlerkorrektur sehr robust ist und in vielen Bereichen Anwendung gefunden hat. Zudem ist dieser gut automatisch im Bild zu detektieren.

Rasterung In der Computergrafik bezeichnet Rasterung die Umwandlung einer Vektorgrafik in eine Rastergrafik, welche sich aus Pixeln zusammensetzt.

Raytracing (Computergrafik) Auch Strahlverfolgung genannt. Es handelt sich um ein Verfahren der Computergrafik um ein Bild aus einer 3D-Szene zu generieren. Dazu wird für jedes Pixel des Sensors der Weg des Lichtes simuliert und verfolgt. Begegnet der Lichtstrahl einem virtuellen Objekt, so werden die auftretenden Effekte simuliert. Durch Anwendung dieser Technik ist ein hoher Grad an visuellem Realismus möglich, welcher jedoch auf Kosten der Rechenzeit geht.

Renderer Modul zum Rendern → rendern von 3D-Grafiken zu 2D-Bildern.

rendern (engl. *(to) render*. Begriff aus der Computergrafik. Rendern beschreibt den Prozess der Umwandlung von Rohdaten zu Bildern. In der Computergrafik handelt es sich oft um virtuelle 3D-Modelle oder ganze 3D-Szenen die in 2D-Bilder umgewandelt werden. Dies ist notwendig, um das Objekt oder die Szene auf einem 2D-Bildschirm anzuzeigen.
Pre-Rendering: Da der Rendering-Prozess sehr rechenintensiv und damit sehr zeitaufwendig sein kann, werden die Bilder zeitunabhängig erstellt und in einer Videodatei gespeichert. Das fertige Video kann schneller abgespielt werden, so dass die Animation flüssig läuft.
Echtzeit-Rendering: Im Gegensatz zum Pre-Rendering hat man beim Echtzeit-Rendering die Möglichkeit, die Animation während der Laufzeit zu modifizieren. Es wird für Computerspiele oder Anwendungen verwendet, bei denen die Animation von zeitkritischen Eingabeparametern abhängt. Im Allgemeinen sind Echtzeit-Animationen nicht so komplex, damit sie schneller gerendert werden können.

RGB Farbraum, der sich durch additive Mischung der drei Grundfarben Rot, Grün und Blau zusammensetzt.

RGB-D-Sensor Sensoren, die neben dem Farbbild ein → Tiefenbild aufzeichnen. Einer der bekanntesten Vertreter ist die Microsoft-Kinect-Kamera.

Rigging (Animation) Verfahren, bei dem einem 3D-Modell ein Skelett bestehend aus → Knochen (Animation) hinzugefügt wird, welches im späteren Verlauf mit dem 3D-Mesh des Modells verbunden wird und zur Animation genutzt werden kann.

ROS-Bag Siehe Kapitel 2.3.2.

ROS-Message Siehe Kapitel 2.3.2.

ROS-Node Siehe Kapitel 2.3.2.

ROS-Service Siehe Kapitel 2.3.2.

ROS-Topic Siehe Kapitel 2.3.2.

Rotoskopie Verfahren zur Animation von Objekten oder Modellen. Es beschreibt den Prozess der Rekonstruktion von Objektbewegung aus einem

Video, indem das virtuelle Gegenstück manuell Bild für Bild angepasst wird bis die Animation mit dem Objektvideo deckungsgleich ist.

Sensel Ist eine von vielen Zellen auf einem bildgebenden Sensor und dient der Aufnahme eines Pixelwertes.

Software-Framework Ein Software-Framework ist ein Programmiergerüst, welches den Software-Entwickler bei der Programmierung unterstützt und Werkzeuge und Funktionen zur Verfügung stellt.

Spiel-Engine engl. *Game-Engine*. Software-Framework für die Entwicklung von Computerspielen. Im Allgemeinen bietet es Werkzeuge für das Design, die Entwicklung und die Ausführung von Spielen. Während der Ausführung rendert (zeichnet) es die virtuelle 3D-Szene in 2D-Bilder zur Darstellung auf einem Anzeigengerät.

Spline-Interpolation Dient der Interpolation von Stützstellen einer Funktion (Argument einer Funktion) durch Polynome niedrigen Grades. Entgegen der Polynominterpolation, bei der es unter Umständen zur Oszillation kommen kann, liefert die Spline-Interpolation meist Kurvenverläufe mit geringer Biegung.

Stimulus (Verhaltensforschung) Im Allgemeinen handelt es sich um einen Reiz, der eine Verhaltensweise auslöst. Bei der Erforschung der Partnerwahl (Fischverhaltensforschung) werden als Stimuli z. B. lebende oder auch künstliche Fische eingesetzt, die z.B. als potenzieller Partner oder Konkurrent dienen.

Szenegraphen-System Ist eine Datenstruktur, die bei der Entwicklung von Computergrafik-Anwendungen wie Videospielen genutzt wird. Szenegraphen-Systeme beinhalten oft räumliche Beziehungen zwischen Objekten in der 3D-Szene und bieten dem Entwickler Werkzeuge und Funktionen zur Bearbeitung und Darstellung der 3D-Szene.

TCP/IP-Protokoll Transmission Control Protocol/Internet Protokoll. Klasse von Netzwerkprotokollen, welche die Kommunikation zwischen Rechnern regeln, die über ein Netzwerk verbundenen sind.

Textur Im Bereich der Computeranimation beschreibt Textur ein Bild, das auf die Oberfläche eines 3D-Objektes oder Modells abgebildet wird.

Tiefenbild Bildet die Entfernung zu einer aufgenommenen Szene in einem Bild ab. Die Farb- oder Helligkeitsinformationen im Bild korrelieren mit der Entfernung zum abgebildeten Objekt.

Topic(ROS) Siehe Kapitel 2.3.2.

Tracking (Video) Ist ein Verfahren zum Lokalisieren und Verfolgen eines sich bewegenden Objekts über die Zeit (Bild für Bild) in einer Videosequenz. Dies kann im 2D- und 3D-Raum erfolgen. Im Falle des 3D-Tracking berechnet das Tracking-System die 3D-Position des Objekts und - falls erforderlich - die 3D-Orientierung.

Treiber Software zum Zugriff auf Hardwarekomponenten im Computersystem.

UV mapping Beschreibt den Prozess der Abbildung einer Textur auf eine Oberfläche eines 3D-Objekts. Um eine 2D-Textur auf einer 3D-Oberfläche zu platzieren, wird die 3D-Objektoberfläche zu einer 2D-Karte aufgeklappt. Da X, Y und Z bereits zur Beschreibung der Achsen des 3D-Objektraums verwendet werden, werden U und V zur Beschreibung der 2D-Koordinaten auf der aufgeklappten Objektoberfläche eingeführt. Die Texturposition bezüglich der Oberflächenkarte wird in U- und V-Koordinaten definiert.

Verhalten (Biologie) Ist eine äußerlich wahrnehmbare Veränderung wie z. B. Bewegung, Geste oder Lautäußerung eines Menschen oder eines Tiere. Als Verhalten kann sowohl die Gesamtheit dieser Vorgänge als auch einzelne über einen bestimmten Zeitraum auftretende Veränderungen bezeichnet werden. In der hier vorliegenden Arbeit werden Bewegungen erst als Verhalten bezeichnet, wenn eine explizite Aktion wie Fressen, Annähern oder Balzen mit dieser in Verbindung steht.

Virtual Reality deutsch virtuelle Realität, kurz VR. Bezeichnet eine Computergenerierte Echtzeit-Animation einer interaktiven, virtuellen Umgebung.

virtuelle Kamera Abstrahiert die zum Rendern (→ rendern) einer Szene notwendigen Parameter wie Blickwinkel und Position und entspricht der Kamera bei einer Videoaufnahme.

weight painting Beschreibt den Prozess der Verbindung des → Mesh (Polygon)s mit den Modellknochen. Da sich nicht alle Teile des Meshs bezüglich eines bestimmten Knochens in gleicher Weise bewegen sollen, kann der Animator mit Hilfe dieser Funktion den relativen Knocheneinfluss auf das zugehörige Mesh definieren.

Weißabgleich Dient der Sensibilisierung der Kamera auf die Farbtemperatur am Aufnahmeort.

YCbCr Farbmodell, welches für das Digitalfernsehen entwickelt wurde. Die Farbinformationen werden in drei Komponenten aufgeteilt: die Grundhelligkeit Y und zwei Farbkomponenten Cb (*Blue-Yellow Chrominance*) und Cr (*Red-Green Chrominance*).

Zustandsautomat Ist eine abstrakte Maschine, die aus Zuständen, Zustandsübergängen und Aktionen besteht. Sie kann zu jeder beliebigen Zeit nur in einem ihrer Zustände sein. Vor oder nach Wechsel eines Zustandes wird meist eine Aktion ausgeführt.

Danksagung

Zum Ende der Arbeit möchte ich mich bei allen bedanken, die mich auf unterschiedliche Weise bei und während der Erstellung unterstützt haben.
Mein erster Dank gilt meinem Doktorvater, Herr Prof. Dr. Kuhnert, der es mir ermöglichte die Dissertation am Institut für Echtzeit Lernsysteme anzufertigen, mir stets mit hilfreichen Tipps zur Seite stand und mir in vielen inspirierenden Gesprächen neue Wege aufzeigte. Die offene und angenehme Arbeitsatmosphäre unter ihm bildete einen optimalen Nährboden für neue Ideen und Projekte auch abseits des *virtuellen Fisches*. Diesen stand er stets offen gegenüber und unterstützte mich bei deren Umsetzung.
Ich möchte mich bei Frau Prof. Dr. Klaudia Witte, die zusammen mit Prof. Kuhnert das Projekt aus der Taufe hob, für die wertschätzende Zusammenarbeit während des gesamten Projektes bedanken. Darüber hinaus danke ich ihr für die Übernahme des Korreferates sowie Herrn Prof. Obermaisser und Herrn Prof. Brück für die Bereitschaft Teil der Prüfungskommission zu sein.
Des Weiteren möchte ich mich bei allen im Projekt beteiligten Personen bedanken, die aktiv und passiv zum Gelingen des *virtuellen Fisches* beigetragen haben. Dazu zählt mein ehemaliger Kollege und Büronachbar Ievgen, der mit neuen Ideen und Entwicklungen das Projekt vorangebracht hat. Mein Dank gilt auch den studentischen Hilfskräften Jan-Marco, ohne den die *FishSim*-Software nicht den heutigen Stand erreicht hätte, und Timo der besonders zu Beginn des Projektes aktiv an der Simulationsumgebung mitgearbeitet hat.
Zudem möchte ich allen im Kooperationsprojekt beteiligten Biologinnen für das stets sehr angenehme und konstruktive Arbeitsklima danken. Mein besonderer Dank geht an Steffi, die mit vielen guten Ideen und Vorschlägen die Entwicklung der *FishSim*-Software aus biologischer Sicht in die richtige Richtung gelenkt und das Projekt auch medienwirksam (z. B. *Molly knows best!*)

der Öffentlichkeit präsentiert hat. Ein weiterer Dank gilt meinen Kollegen und ehemaligen Kollegen Duong, Ievgen, Jan, Jens, Khaled, Lars, Marc Steven, Markus, Matthias, Matthias, Massoud, Nazeer, Salih, Sina und Tao. Ich habe mich auch dank eurer Anwesenheit immer sehr wohl am Institut gefühlt. So mancher erfolglose Vormittag wurde durch die erfrischende Mensarunde mit inspirierenden Gesprächen wettgemacht und der Tag in neue Bahnen gelenkt. Unvergesslich sind auch die Kaffeerunden, die stets ein Feuerwerk an Ideen mit sich brachten.

Ein großer Dank gilt meiner Familie – meinen Eltern, Geschwistern und Schwiegereltern. Während der Promotion waren Sie stets da, um mir den Rücken frei zu halten wenn eine Deadline näher rückte, unsere Kinder krank waren und fürsorgliche Pflege benötigten oder sonst Not am Mann war.

Mein letzter Dank gilt meiner Frau und meinen Kindern. Kathi war dar, wenn es an Struktur fehlte, half mir mich auf das Wesentliche zu konzentrieren, baute auf wenn etwas nicht lief und stand mir als verlässliche und gewissenhafte Korrektorin nicht nur bei dieser Arbeit zur Seite. Kathi, vielen Dank für alles! Meine Kinder sorgten auf wunderbare Weise dafür, dass die Dissertation mein Leben nicht komplett eingenommen hat.