

**Modellierung von Schwarmverhalten,
Untersuchung von Methoden zur Parameter
Approximation**

Simon Christofzik

Konstanz, 8. Dezember 2021

MASTERARBEIT

MASTERARBEIT

zur Erlangung des akademischen Grades

Master of Science (M. Sc.)

an der

Hochschule Konstanz

Technik, Wirtschaft und Gestaltung

Fakultät Informatik

Studiengang Master of Science Informatik

Thema:

Modellierung von Schwarmverhalten, Untersuchung von Methoden zur Parameter Approximation

Masterkandidat:

Simon Christofzik
Engelbrechtstr 39
78234 Engen

1. Prüfer:

Prof. Dr. Rebekka Axthelm

2. Prüfer:

Prof. Dr. Oliver Dürr

Ausgabedatum:

5. Mai 2021

Abgabedatum:

8. Dezember 2021

Ehrenwörtliche Erklärung

Hiermit erkläre ich *Simon Christofzik, geboren am 01.11.1989 in Engen,*

- (1) dass ich meine Masterarbeit mit dem Titel

Modellierung von Schwarmverhalten, Untersuchung von Methoden zur Parameter Approximation

am Institut für Optische Systeme unter Anleitung von Prof. Dr. Rebekka Axthelm selbstständig und ohne fremde Hilfe angefertigt habe und keine anderen als die angeführten Hilfen benutzt habe;

- (2) dass ich die Übernahme wörtlicher Zitate, von Tabellen, Zeichnungen, Bildern und Programmen aus der Literatur oder anderen Quellen (Internet) sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe.
(3) dass die eingereichten Abgabe-Exemplare in Papierform und im PDF-Format vollständig übereinstimmen.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Konstanz, 8. Dezember 2021

(Unterschrift)

Abstract

Titel:	Modellierung von Schwarmverhalten, Untersuchung von Methoden zur Parameter Approximation
Masterkandidat:	Simon Christofzik
Prüfer:	Hochschule Konstanz Technik, Wirtschaft und Gestaltung Institut für Optische Systeme
	Prof. Dr. Rebekka Axthelm
	Prof. Dr. Oliver Dürr
Abgabedatum:	8. Dezember 2021
Schlagworte:	Boids, Collective behaviour, Reverse mode differentiation, Particle swarm optimization

In dieser Arbeit wird mithilfe von Optimierungsverfahren Parameter für Schwarmmodelle geschätzt. In der gängigen Literatur zu Schwarmverhalten stößt man immer wieder auf agentenbasierte Modelle. Eines der bekanntesten Modelle im Bezug auf agentenbasierte Modellierung ist das Boidsmodell, welches von Craig Reynolds vor über 30 Jahren publiziert wurde. Ziel dieser Arbeit ist es, Zustände, welche in echten Schwärmen zu beobachten sind, durch das Boidsmodell zu imitieren. Dazu werden die Parameter des Schwarmmodells an Datensätze von Schwärmen angepasst. Des Weiteren wird ein neues Modell vorgestellt, welches im Laufe der Arbeit entstanden ist. Dieses Modell wird ebenfalls per Approximation der Parameter an die Datensätze angepasst. Der direkte Vergleich der erreichten Zustände der Modelle und der Zustände des realen Datensatzes gibt hierbei Aufschluss über die Modellannahmen.

Extended Abstract

Titel:	Modellierung von Schwarmverhalten, Untersuchung von Methoden zur Parameter Approximation
Masterkandidat:	Simon Christofzik
Prüfer:	Hochschule Konstanz Technik, Wirtschaft und Gestaltung Institut für Optische Systeme Prof. Dr. Rebekka Axthelm Prof. Dr. Oliver Dürr
Abgabedatum:	8. Dezember 2021
Schlagworte:	Boids, Collective behaviour, Reverse mode differentiation, Particle swarm optimization

Einleitung

Schwarmverhalten ist eine der grundlegenden Verhaltensmuster in unserer Welt. Aufgaben, die für das Individuum unmöglich sind, werden durch die Zusammenarbeit vieler Individuen erst möglich. Das Verständnis über das Verhalten von Schwärmen kann in vielen technischen Bereichen Fortschritt bedeuten. Beispielsweise kann das autonome Fahren, bei dem jedes Fahrzeug individuell Entscheidungen trifft, von Schwarmmodellen profitieren.

In dieser Arbeit wird sich mit zwei Schwarmmodellen auseinandergesetzt. Das Ziel ist, mit Hilfe eines Modells Verhaltensmuster eines echten Schwarms abzubilden, indem Parameter für die vorgestellten Modelle geschätzt werden. Dazu wird das Modell auf Realdaten justiert und das Verhalten des Modells dem des echten Schwarms gegenübergestellt.

Methodik

Um Parameter anhand von Realdaten zu approximieren, werden Trajektorien von Fischschwärmern unterschiedlicher Schwarmgröße verwendet. Die Daten hierzu sind unter <https://idtracker.ai> zu finden.

Als Modell wird ein Boidsmodell [1] und ein eigenes Modell betrachtet. Die Parameter der Modelle werden für das Boidsmodell mit dem Verfahren des Particle Swarm Optimization (PSO) approximiert [4]. Für das eigene Modell

wird Reverse Mode Differentiation (RMD) [2] verwendet. Die Verhaltensmuster, welche abgebildet werden sollen, sind die Polarisierung und die Rotation des Schwarmes [3]

Ergebnisse und Zusammenfassung

Die Approximation der Parameter hat gezeigt, dass die Schwarmzustände für das eigene Modell abgebildet werden können.

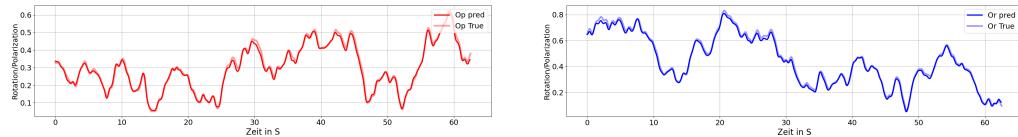


Abbildung 1: Polarisationszustand (links) und Rotationszustand (rechts) der Realdaten mit 60 Fischen und des eigenen Modells

Für das Boidsmodell war es nicht möglich die Zustände zu imitieren.

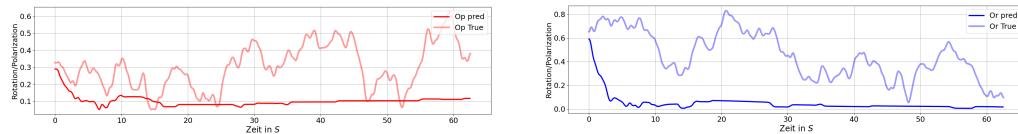


Abbildung 2: Polarisationszustand (links) und Rotationszustand (rechts) der Realdaten mit 60 Fischen und des metrische Boids Modells

Auch die Trajektorien zeigen, dass die Approximation für das Boidsmodell in einem unrealistischen Szenario endet. Das eigene Modell zeigt eine klare Präferenz hinsichtlich der Bewegung und ist ebenfalls vom echten Schwarm zu unterscheiden.



Abbildung 3: Trajektorien der Realdaten für 60 Fische (links), des metrischen Boids Modells (mitte) und des eigenen Modells (rechts)

Literatur

- [1] Iain D. Couzin u. a. „Collective memory and spatial sorting in animal groups.“ In: *Journal of theoretical biology* 218 1 (2002), S. 1–11.
- [2] Charles C. Margossian. „A Review of automatic differentiation and its efficient implementation“. In: *CoRR* abs/1811.05031 (2018). arXiv: 1811.05031. URL: <http://arxiv.org/abs/1811.05031>.
- [3] Kolbjørn Tunstrøm u. a. „Collective States, Multistability and Transitional Behavior in Schooling Fish“. In: *PLoS Computational Biology* 9 (2013).
- [4] Dongshu Wang, Dapei Tan und Lei Liu. „Particle swarm optimization algorithm: an overview“. In: *Soft Computing* 22 (Jan. 2018). DOI: 10.1007/s00500-016-2474-6.

Danksagung

An dieser Stelle möchte ich mich herzlich bei Prof. Dr. Rebekka Axthelm und Prof. Dr. Oliver Dürr für das Ermöglichen dieser Arbeit bedanken. Insbesondere bin ich dankbar für die gute Betreuung von Frau Axthelm, die sich sehr viel Zeit für meine Fragen nahm und nahezu rund um die Uhr Antworten auf diese geben konnte.

Inhaltsverzeichnis

Ehrenwörtliche Erklärung	I
Abstract	II
Extended Abstract	III
Danksagung	VI
1 Einleitung	1
2 Grundlagen	3
2.1 Schwarmverhalten	3
2.1.1 Boids	3
2.2 Kombinatorische Optimierung	4
2.3 Gradientenabstieg	5
2.3.1 Reverse mode differentiation	6
2.4 Particle swarm optimisation	8
2.5 Optischer Fluss	11
3 Stand der Forschung	14
3.1 Zustände von Schwärmen	14
3.2 Modelle	15
3.2.1 Metrisches Modell	15
3.2.2 Eigenes Modell	17
4 Experimente	21
4.1 Tracking	21
4.1.1 Eigene Trackingmethode	22
4.1.2 Tracking via Neural Network	25
4.1.3 Eigenschaften des Datensatzes	27
4.2 Approximation Anhand künstlich erzeugter Daten	30

4.2.1	Vergleich zwischen PSO und RMD	30
4.2.2	Approximierung von konstanten Parametern	32
4.2.2.1	Metrisches Boids Modell	32
4.2.2.2	Eigenes Modell	37
4.2.3	Diskussion	40
4.2.4	Approximierung von variablen Parametern	40
4.2.4.1	Metrisches Boids Modell	41
4.2.4.2	Eigenes Modell	44
4.2.5	Diskussion	47
4.3	Approximation anhand von realen Daten	48
4.3.1	Diskussion	54
5	Fazit und Ausblick	55
Literatur		VIII
Abbildungsverzeichnis		X
Tabellenverzeichnis		XI
Anhänge		XII
A Weitere Abbildungen		XIII

1 Einleitung

Schwarmverhalten ist eine der grundlegenden Verhaltensmuster in unserer Welt. Aufgaben, die für das Individuum unmöglich sind, werden durch die Zusammenarbeit vieler Individuen erst möglich. Faszinierend ist, dass der einzelne Teilnehmer des Schwarms das große Ganze nicht sieht, durch die Kooperation dennoch etwas Großes zustande kommt. Einfache Lebensformen wie beispielsweise Ameisen verbringen im Kollektiv bemerkenswerte Aufgaben. Das Nest verschiedener Ameisenarten sind kunstvoll in ihrer Architektur und dennoch funktional. Die Blattschneiderameise *Atta vollenweideri* errichtet Bauten, deren Belüftung den Gehalt von Kohlenstoffdioxid im Bau reguliert. Der Ameisenbau beinhaltet Kammern, in denen Nahrungsmittel wie Pilze heranwachsen [7]. Es ist bemerkenswert, dass diese Kreaturen auf der Basis von lokalen Informationen und selbstständiger Arbeitsweise Aufgaben vollbringen, ohne eine zentrale Steuereinheit zu besitzen. Auch in uns Menschen stecken Systeme, die nach dem Muster des Schwarmverhaltens schützen und somit am Leben halten, wie zum Beispiel unser Immunsystem. Neutrophils spielen eine große Rolle in der frühen Bekämpfung von Krankheitserregern. Durch das Aussenden von Chemotaxis wird die Immunabwehr angeregt und koordiniert dadurch das Schwarmverhalten von Nachbarzellen [9].

Die Bionik beschäftigt sich mit dem Übertragen von Phänomenen aus der Natur in die Technik. Eines der bekanntesten Beispiels ist das der Gebrüder Wright, die durch das beobachten von Vögeln ihre Flugzeugprototypen optimierten [19]. Auch die zuvor erwähnte Ameisenkollonie diente bei der Entwicklung des Ameisenalgorithmus als Vorbild für die Suche nach einem kürzesten Weg in einem Graphen.

In den vergangenen Jahren ist die Entwicklung des autonomen Fahrens immer stärker vorangeschritten. Das autonome Fahren verspricht viele Vorteile, wie das selbstständige Transportieren von Personen, wodurch diese nicht mit Fahren beschäftigt sind und sich produktiveren Angelegenheiten widmen können. Auch effizientere Fahrtwege und das vermeiden von Staus können daraus hervorgehen. Hierdurch kann ein geringerer Kraftstoffverbrauch resultieren, was unserem Klima zu gute kommen kann. Die Bezeichnung autonomes Fahren impliziert, dass Autos keiner zentrale Steuereinheit folgen, sondern selbstständig Entscheidungen treffen. Diese müssen im Einklang mit den lokalen Einflüssen wie Wetterbedingungen, Fahrbahnbeschaffenheit und insbesondere der anderen Verkehrsteilnehmer getroffen werden. Doch wie kann das autonome Fahren effizient und sicher gestaltet werden? Die Antwort scheint wie so oft in der Tierwelt zu liegen und dem Studieren von Schwarmverhalten.

Das Verständnis von Phänomenen der Natur wird üblicherweise durch Modelle geschaffen, die das Phänomen erklären sollen. Schwarmverhalten stellt hier keine Ausnahme dar. Durch das Formulieren von Schwarmmodellen können

Eigenschaften eines Schwarms genauer betrachtet werden und auch Vorhersagen über das Verhalten eines Schwarms gemacht werden.

In dieser Arbeit wird sich mit einem gängigen Schwarmmodell auseinandersetzt. Das Ziel ist, mit Hilfe eines Modells Verhaltensmuster eines echten Schwarms abzubilden, indem Parameter für die vorgestellten Modelle geschätzt werden. Dazu wird das Modell auf Realdaten justiert und das Verhalten des Modells dem des echten Schwarms gegenübergestellt. Aus den Ideen und Beobachtungen des gängigen Modells wird zudem ein eigenes Modell abgeleitet. Dadurch können zwei Modelle gegenübergestellt und verglichen werden.

Der Aufbau der Arbeit beginnt mit den Grundlagen, die notwendig sind, um dieses Unterfangen zu bewältigen. Die grundlegenden Eigenschaften eines Schwarms werden zu Beginn angeführt. Darauf folgen Optimierungsverfahren, welche es ermöglichen, Daten und Modell in Einklang zu bringen. Anschließend folgt ein Exkurs über die Betrachtung von Bewegungen, wodurch Videoaufnahmen von Schwärmen in eine Form gebracht werden, um diese verarbeiten zu können. Die mathematische Definition der Modelle wird in dem darauffolgenden Kapitel eingeführt. Zudem folgt die Definition der Zustände, die dem Datensatz innewohnen. Das vorletzte Kapitel widmet sich den Experimenten, in denen die Pfade der Realdaten extrahiert werden und die Modelle an diese angepasst werden. Schlussendlich werden die Ergebnisse im Ausblick für weitere Arbeiten dargestellt.

2 Grundlagen

Dieses Kapitel widmet sich den Grundlagen für diese Arbeit. Hierzu gehört die Definition der Modellannahmen. Des Weiteren werden Optimierungsverfahren vorgestellt. Der Schlussteil des Kapitels widmet sich dem optischen Fluss, durch den Bewegungen geschätzt werden.

2.1 Schwarmverhalten

Die Verhaltensbiologie unterscheidet die Begriffe Verhaltensweisen und Verhaltensmuster. Ersteres sind voneinander unterscheidbare und beobachtbare Bewegungsabläufe. Das sind beispielsweise Bewegung des Körpers oder Körperhaltung. Das Verhaltensmuster ist durch beobachtbare Abfolgen von Verhaltensweisen definiert. Schwarmverhalten wird als komplexes Verhaltensmuster definiert [8]

2.1.1 Boids

Boids ist ein Modell, welches von Reynolds 1987 veröffentlicht wurde. Seine Intention war es, ein Modell zu präsentieren, welches Schwarmverhalten simuliert, um dies für Animationen nutzen zu können. Die Grundvoraussetzungen für einen Schwarm sind das Zusammenhalten der Individuen, eine gemeinsame Bewegungsrichtung und das Vermeiden von zu geringen Abständen zwischen den Individuen. Um dies zu erreichen, setzt Reynolds auf ein agentenbasiertes Modell, d.h. jeder Agent im Schwarm handelt individuell. Der Schwarm wird dadurch zu einer Einheit, die sich selbst organisieren kann, ohne durch eine zentrale Stelle gesteuert zu werden. Ursprünglich formulierte Reynolds für seine Agenten drei zentrale Theoreme.

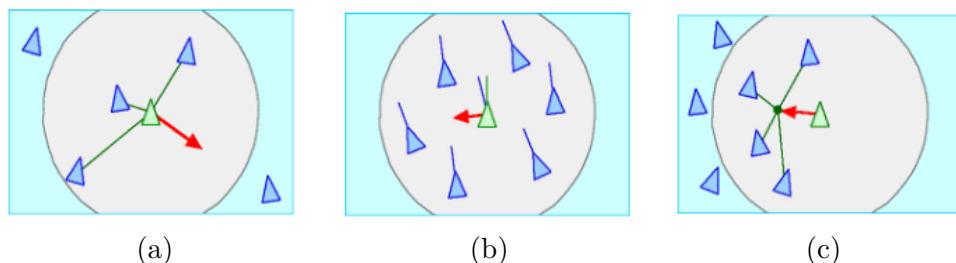


Abbildung 2.1: Verhaltensmuster nach Reynolds, entnommen aus [15]

a) Abstandsregel

Jeder Agent versucht das Zusammenstoßen mit anderen Agenten zu vermeiden. Hierbei betrachtet der Agent alle Agenten in einem bestimmten Radius R . In

Abbildung 2.1 (a) ist zu sehen wie sich der Agent in Grün von den anderen Agenten innerhalb der Zone wegbewegt. Dies hat zur Folge, dass die Dichte des Schwarms nicht zu groß wird.

b) Angleichen

Der Agent möchte in die gleiche Richtung schwimmen wie Agenten in einer definierten Zone. Dadurch agiert das Kollektiv als eine Einheit und bewegt sich in dieselbe Richtung.

c) Zusammenbleiben

Der Agent hat das Ziel, beim Schwarm zu bleiben. Dazu schwimmt er in Richtung des Massezentrums welches durch die Agenten in der Zone entsteht.

Zu beachten ist, dass Reynolds in seinem Paper nur die Idee für ein Modell gibt. Ein expliziter Algorithmus wird jedoch nicht definiert Reynolds [16].

Das Modell nach Reynolds ist rein metrisch, jedoch hat eine Studie von Ballerini Ballerini u. a. [1] gezeigt, dass insbesondere Stare nur die sechs bis sieben nächsten Nachbarn, unabhängig von deren Distanz wahrnehmen. Dies wird als topologischer Ansatz bezeichnet.

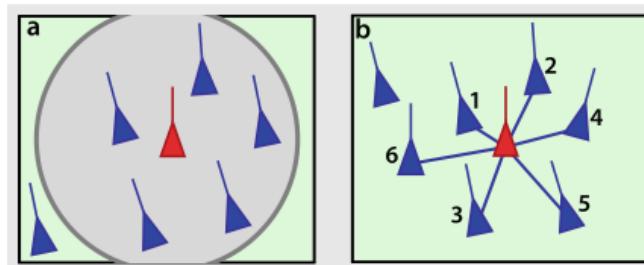


Abbildung 2.2: Metrisch und Topologische Distanzen, entnommen aus [8]

Insofern kann das Modell von Reynolds auch durch den topologischen Ansatz formuliert werden. Hierbei werden die drei Zonen auf die n -nächsten Nachbarn aufgeteilt. Der Vorteil an diesem Modell ist, dass der Kontakt zum Schwarm nie verloren geht. Beim metrischen Modell ist es möglich, dass sich ein Agent von dem Schwarm so weit entfernt, dass kein Kontakt mehr zu anderen Agenten besteht. Das topologische Modell hingegen gewährleistet stets eine Verbindung zum Kollektiv. Das eben beschriebene Szenario tritt hauptsächlich bei plötzlichen externen Ereignissen auf. Beispielsweise ist ein Angriff auf ein Kollektiv so ein Szenario, bei dem einzelne Agenten vom Schwarm getrennt werden. Individuen im Schwarm gelten jedoch als stärker, als solche die sich vom Schwarm getrennt haben. Daher ist das Ziel eines Schwarms das Zusammenbleiben, was robuster durch das topologische Modell als durch das metrische Modell erreicht wird.

2.2 Kombinatorische Optimierung

Kombinatorische Optimierung bezeichnet das Auffinden einer oder mehrerer Teilmengen aus einer Grundmenge an Variablen, welche bezüglich einer Kos-

tenfunktion maximal oder minimal sind. Kombinatorische Optimierung ist ein relevantes Forschungsthema und hat durch Machine Learning und auch insbesondere durch künstliche neuronale Netze einen immer größer werdende Relevanz. Pankaj K. Agarwal [13]

Mathematisch kann man ein Optimierungsproblem bezüglich einer Kostenfunktion folgendermaßen definieren:

Sei $f : \mathbb{R}^D \rightarrow \mathbb{R}$ eine Fehlerfunktion.

Dann wird \hat{x} gesucht, so dass $f(\hat{x}) \geq f(x) \quad \forall x \in \mathbb{R}^D$

Analog für die Suche des Minimums.

2.3 Gradientenabstieg

Der Gradientenabstieg ist eine Methode, um das Minimum oder auch Maximum einer Funktion zu berechnen. Hierbei wird über die Steigung an einer Stelle ermittelt, in welche Richtung die Extremstelle liegt. Wenn kein Vorwissen über die Position der Extremstelle besteht, wird die Startposition zufällig gewählt. Von dieser Position aus wird nun die Steigung bestimmt, um sich in Richtung der Extremstelle zu bewegen.

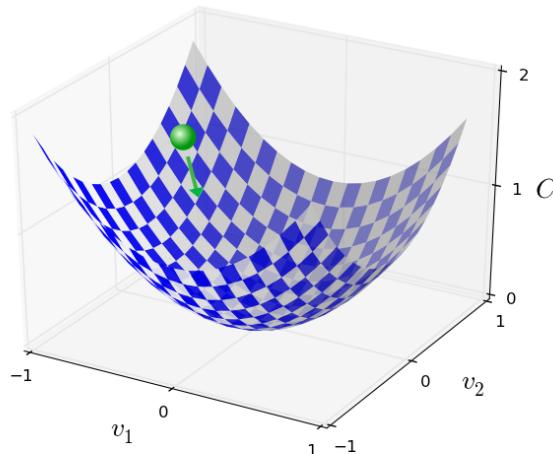


Abbildung 2.3: Gradientenabstieg visualisiert entnommen von <http://neuralnetworksanddeeplearning.com/chap1.html> besucht am 24.11.2021

Abbildung 2.3 visualisiert den Gradientenabstieg. Der grüne Ball symbolisiert die Position an der die Steigung berechnet wird. Der Pfeil stellt die Schrittweite des Abstiegs dar. Die Schrittweite welche auch als Lernrate bezeichnet wird, ist ein kritischer Aspekt. Wird diese zu groß gewählt, so oszilliert der Ball um

die Extremstelle und erreicht diese eventuell nicht. Ist diese hingegen zu klein, so dauert der Abstieg unverhältnismäßig lange. Ein Verfahren welches dieses Problem löst nennt sich *ADAM* und steht für adaptive moment estimation. ADAM ist eine Methode um die Lernrate anzupassen und wird häufig als Alternative zum Stochastic Gradient Descent (SGD) im Deep Learning Bereich verwendet. Die Regel für das Anpassen der Position lautet:

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \quad (2.1a)$$

$$n_t = \beta_2 \cdot n_{t-1} + (1 - \beta_2) \cdot g_t^2 \quad (2.1b)$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \quad (2.1c)$$

$$\hat{n}_t = \frac{n_t}{1 - \beta_2^t} \quad (2.1d)$$

$$v_t = v_{t-1} - \lambda \cdot \frac{\hat{m}_t}{\sqrt{\hat{n}_t + \epsilon}} \quad (2.1e)$$

In Gleichung 2.1a und 2.1b bezeichnet g_t den berechneten Gradienten an einer gegebenen Stelle. Initialisiert wird m_0 und n_0 mit 0. Die Parameter β_1 und β_2 werden mit 0.9 und 0.999 angegeben. Die Variablen m_t und n_t bringen die vergangenen Gradienten in den Updateprozess mit ein. Dadurch fallen plötzliche Richtungsänderungen weniger dramatisch aus. Der Parameter ϵ ist eine Konstante und wird mit $\epsilon = 10^{-8}$ angegeben. λ ist die Lernrate und v_t die schlussendliche Schrittrichtung Diederik P. Kingma [3].

Zu erwähnen ist, dass beim Gradientenabstieg nicht sichergestellt werden kann, dass das Minimum auch ein globales und nicht nur ein lokales Minimum ist (gleiches gilt für das Maximum). Ausgenommen hiervon sind konvexe Optimierungsprobleme. Bei dieser Art ist jedes lokale Minimum/Maximum auch ein Globales.

2.3.1 Reverse mode differentiation

Das Berechnen von Ableitungen hat im Bereich der Optimierung einen sehr hohen Stellenwert. Für viele Problemstellungen können Ableitungen nicht analytisch berechnet werden, lediglich die Auswertung an bestimmten Stellen ist möglich. Die Ableitung an einer bestimmten Stelle ist essenziell, um den Gradientenabstieg zu ermöglichen. Für eine differenzierbare Funktion, welche in numerischer Form vorliegt, ist Reverse Mode Differentiation (RMD) ein Verfahren, um deren Ableitung zu berechnen. Grundbaustein hierfür ist die Kettenregel [11].

Betrachten wir eine Funktion $z = f(f_1(t), f_2(t))$, welche von zwei weiteren Funktionen f_1 und f_2 abhängt. Die Funktionen f_1 und f_2 hängen wiederum von t ab. Dafür sieht die Kettenregel hierfür wie folgt aus:

$$\frac{\delta z}{\delta t} = \frac{\delta z}{\delta f_1} \frac{\delta f_1}{\delta t} + \frac{\delta z}{\delta f_2} \frac{\delta f_2}{\delta t} \quad (2.2)$$

Dies kann grafisch folgendermaßen dargestellt werden:

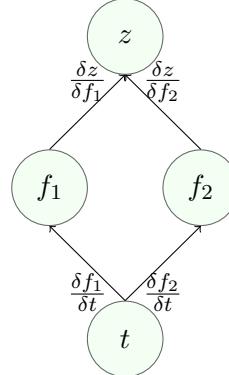


Abbildung 2.4: Kettenregel als Graph

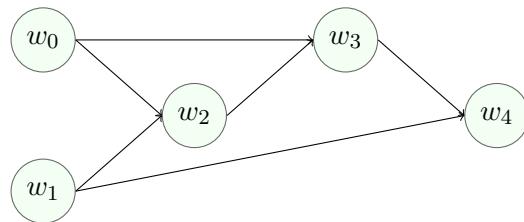
Die grafische Darstellung einer Funktion, wie sie in Abbildung 2.4 zusehen ist, kann für jede differenzierbare Funktion angewandt werden. Folgt man dem Graphen "rückwärts" von oben nach unten, erhält man Einsicht über die Abhängigkeiten der Ableitung. Es kann also mittels Kettenregel die Ableitung einer Funktion an jeder Stelle berechnet werden. Nachfolgend ist ein Beispiel zu sehen, welches dies veranschaulicht.

Betrachten wir folgende Funktion:

$$f(x_1, x_2) = x_1 + x_2 \sin(x_1) + \ln(x_2) \quad (2.3)$$

Dies lässt sich in komponentenschreibweise ausdrücken, wodurch sich folgender Graph ergibt:

$$\begin{aligned} w_0 &= x_1 \\ w_1 &= x_2 \\ w_2 &= w_1 \sin(w_0) \\ w_3 &= w_0 + w_2 \\ w_4 &= z = w_3 + \ln(w_1) \end{aligned}$$



Soll nun die Ableitung an der Position $x_1 = 2, x_2 = 3$ bestimmt werden, so wird der sogenannte *Forwardpass* durchlaufen. Hierbei wird der Graph vom Start bis zum Ende durchlaufen, wobei auf dem Weg die nötigen Ableitungen berechnet werden können.

$$w_0 = 2$$

$$w_1 = 3$$

$$\begin{array}{lll} w_2 = w_1 \sin(w_0) & \frac{\delta w_2}{\delta w_1} = \sin(w_0) = 0.909 & \frac{\delta w_2}{\delta w_0} = w_1 \cos(w_0) = -1.24 \\ w_3 = w_1 + w_2 & \frac{\delta w_3}{\delta w_1} = 1 & \frac{\delta w_3}{\delta w_2} = 1 \\ w_4 = z = w_3 + \ln(w_1) & \frac{\delta w_4}{\delta w_3} = 1 & \frac{\delta w_4}{\delta w_1} = \frac{1}{w_1} = \frac{1}{3} \end{array}$$

Mit Hilfe der Kettenregel kann nun $\frac{\delta w_4}{\delta w_0}$ und $\frac{\delta w_4}{\delta w_1}$ berechnet werden. Der Graph und der *Forwardpass* geben Aufschluss über die Abhängigkeiten der Ableitungen sowie dessen Werte. Die Ableitungen werden folgendermaßen berechnet:

$$\begin{aligned} \frac{\delta w_4}{\delta w_0} &= \frac{\delta w_4}{\delta w_3} \frac{\delta w_3}{\delta w_2} \frac{\delta w_2}{\delta w_0} + \frac{\delta w_4}{\delta w_3} \frac{\delta w_3}{\delta w_0} &= 1 \cdot 1 - 1.24 + 1 \cdot 1 = -0.24 \\ \frac{\delta w_4}{\delta w_1} &= \frac{\delta w_4}{\delta w_3} \frac{\delta w_3}{\delta w_2} \frac{\delta w_2}{\delta w_1} + \frac{\delta w_4}{\delta w_1} &= 1 \cdot 1 \cdot 0.909 + \frac{1}{3} = 1.242 \end{aligned}$$

Das Rückwärtslaufen auf dem Graphen wird auch als *Backwardpass* bezeichnet. Das Resultat ist die Ableitung an der Stelle (2,3) und kann für den Gradientenabstieg genutzt werden.

2.4 Particle swarm optimisation

Particle swarm optimization (PSO) ist ein Verfahren, um optimale Parameter eines Optimierungsproblems zu finden. Es wurde 1995 von Russell C. Eberhart und James Kennedy publiziert. Der Algorithmus wird als Metaheuristik eingestuft, d.h. es ist nicht garantiert, dass eine optimale Lösung gefunden wird. Prinzipiell kann jedoch eine Lösung gefunden werden. Während beim Gradientenabstieg die Steigung der Kostenfunktion entscheidend ist, kommt PSO ganz ohne Ableitung der Kostenfunktion aus. Der Vorteil ist, dass das Optimum einer nicht ableitbaren Funktion bestimmt werden kann.

Wie der Name PSO vermuten lässt, handelt es sich um einen Algorithmus, der auf Schwarmverhalten setzt. Jeder einzelne Agent wird als Partikel bezeichnet, dessen Position im D-dimensionalen Raum eine potenzielle Lösung des Optimierungsproblems darstellt. Die Partikel tasten den Raum Schritt für Schritt ab, um so iterativ den optimalen Parameter zu finden. In jedem Iterationsschritt evaluieren die Partikel ihre Position anhand einer Kostenfunktion. Der beste erreichte Wert und dessen Position steht jedem Partikel zu jedem Zeitschritt zur Verfügung. Somit hat jeder Partikel zu jedem Zeitpunkt einen besten persönlichen Optimum (lokales Optimum) und hat zudem Kenntnisse über das

beste globale Optimum (Bester erreichter Wert aller Partikel zu allen vorherigen Zeitpunkten). Demnach können sich die Partikel (je nach Konfiguration) in dessen Richtung bewegen oder weiter ihr lokalen Bestwert verbessern. Der Ablauf dieses Algorithmus kann somit als ein Rennen um die besten Parameter betrachtet werden. Die Partikel möchten sich, solange kein Abbruchkriterium eingetreten ist, in dieser Hinsicht übertrumpfen.

Der Schwarm der GröÙe N ist definiert durch seine Partikel p_i :

$$X = [p_1, p_2, \dots, p_N] \quad (2.7)$$

Jeder Partikel besitzt eine Position die durch den D-Dimensionalen Vektor p_i definiert ist:

$$p_i = [x_{i1}, x_{i2}, \dots, x_{iD}] \quad (2.8)$$

In jedem Iterationsschritt wird die neue Position $p_i(t+1)$ des Partikels berechnet, indem auf die derzeitige Position $p_i(t)$ ein Richtungsvektor $v_i(t+1)$ addiert wird.

$$p_i(t+1) = p_i(t) + v_i(t+1) \quad (2.9)$$

Der Richtungsvektor $v_i(t+1)$ setzt sich zusammen aus dem vorherigen Richtungsvektor $v_i(t)$, dem Vektor der in Richtung des persönlich besten Wertes zeigt ($pb_i - p_i$) und dem Vektor der in Richtung des globalen Bestwertes zeigt ($gb - p_i$):

$$v_i(t+1) = v_i(t) + c_1(pb_i - p_i)R_1 + c_2 * (gb - p_i)R_2 \quad (2.10)$$

Das Buchführen über die Richtungsvektoren $v_i(t)$ des vorherigen Iterationsschrittes stellt sicher, dass beim Berechnen der neuen Richtung kein drastischer Richtungswechsel stattfindet. Die Konstanten c_1 und c_2 werden als "cognitive coefficient" und "social coefficient" bezeichnet. Sie bestimmen, wie stark in Richtung des persönlichen bzw. globalen Bestwerts gesteuert werden soll. Üblicherweise wird der Bereich von c_1 mit $0 \leq c_1 \leq 4$ angegeben und der Bereich von c_2 wird mit $c_2 \leq 4$. Generell wird es als ausreichend angesehen, $c_1 = c_2 = 2$ zu setzen (vgl. [12]). Die Variablen R_1 und R_2 sind Zufallszahlen, welche aus einer auf $[0, 1]$ verteilten Gleichverteilung gezogen werden. Dadurch bekommen die Richtungen einen statistischen Einfluss.

Velocity explosion bezeichnet das Unkontrollierte größer werden des Richtungsvektors v_i . Die Gewichtung der Richtungsvektoren in Gleichung 2.10 mit den Konstanten c_1 und c_2 sowie den Zufallszahlen R_1 und R_2 hat zur Folge, dass das Ziel in 50% der Fälle überschritten wird. Dies ist der Tatsache geschuldet, dass R_1 und R_2 aus einer Gleichverteilung über $[0, 1]$ gezogen werden. Durch die Multiplikation mit c_1 bzw. c_2 ändert sich die Gleichverteilung auf $[0, 2]$. Um dies zu vermeiden wird eine Gewichtung ω eingeführt (auch als *inherita weight*

bezeichnet). Die Publikation von Shi und Obaiahnabatti [18] zeigt, dass das *inherita weight* im Intervall $[0.9, 1.2]$ liegen sollte.

Die neue Aktualisierungsregel inklusive *inherita weight* ist in folgender Gleichung zu sehen.

$$v_i(t + 1) = \omega(t + 1)v(t) + c_1(pb_i - p_i)R_1 + c_2 * (gb - p_i)R_2 \quad (2.11)$$

Der PSO-Algorithmus wird durch das nachfolgende Flowchart beschrieben.

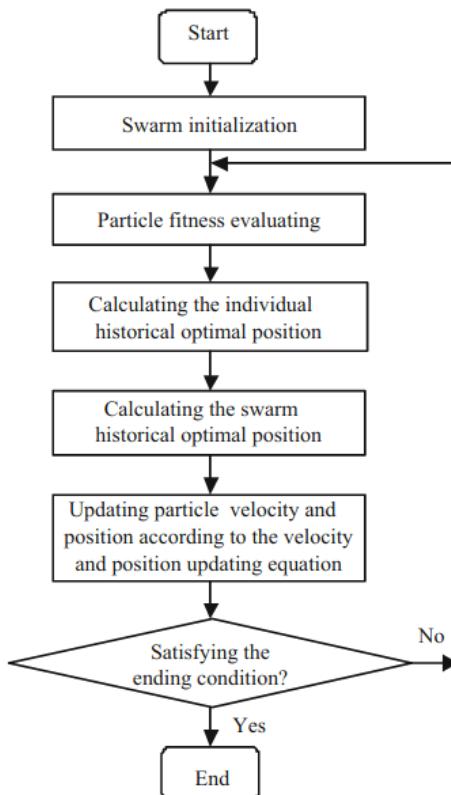


Abbildung 2.5: Flowchart des PSO entnommen aus [21]

Zu Beginn werden die Positionen der Partikel im D-dimensionalen Suchraum aus einer Gleichverteilung gezogen. Hierbei sind sich viele Autoren einig, dass dies den Suchraum am besten abdeckt. Daraufhin wird der Fehler der Positionen durch die Kostenfunktion berechnet. Dies ist initial der persönliche Bestwert jedes Partikels. Zudem wird hieraus der globale Bestwert bestimmt. Es folgt die Berechnung des neuen Richtungsvektors $v_i(t + 1)$ der Partikel (vgl. 2.11). Daraufhin erfolgt die Berechnung der neuen Position. Wird ein Abbruchkriterium erreicht, wie z. B. das Unterschreiten einer Fehlertoleranz oder das Erreichen einer Anzahl an Iterationen, so wird der Algorithmus beendet. Andernfalls beginnt die nächste Iteration [12].

2.5 Optischer Fluss

Das Ziel beim optischen Fluss ist es, die Bewegung von Objekten innerhalb zweier aufeinanderfolgenden Sequenzen zu berechnen. Die Bewegung ist in diesem Szenario relativ zum Betrachter (z. B. Kamera) der Szene. Das Konzept des optischen Flusses wurde 1950 von dem Psychologen James J. Gibson in dem Buch "The perception of the visual world" erstmals diskutiert Gibson [6].

Die grundlegende Idee des optischen Flusses ist es, den Vektor der Bewegung $\delta = [\delta_x, \delta_y]^T$ eines Pixels p zwischen zwei Zeitpunkten zu berechnen.

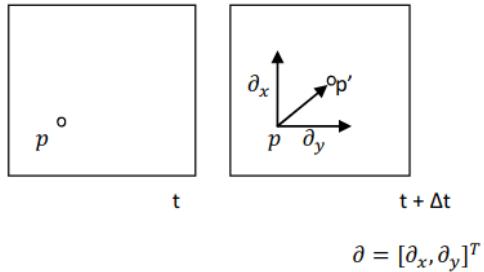


Abbildung 2.6: Bewegung des Pixels p von Zeitpunkt t zu Zeitpunkt $t + \Delta t$ entnommen aus [14]

Insbesondere kann man annehmen, dass der Pixel $I(x, y, t)$ des Bildes zum Zeitpunkt t der selbe ist, wie der Pixel p' des Bildes zum Zeitpunkt $t + \Delta t$ verschoben durch δ . Formal gilt also:

$$I(x, y, t) = I(x + \delta x, y + \delta y, t + \delta t) \quad (2.12)$$

Hier kann nun eine Taylor-Reihenentwicklung angewendet werden, wodurch sich folgende Gleichung ergibt:

$$I(x + \delta x, y + \delta y, t + \delta t) = I(x, y, t) + \frac{\delta I}{\delta X} \delta x + \frac{\delta I}{\delta Y} \delta y + \frac{\delta I}{\delta t} \delta t + H.O.T. \quad (2.13)$$

Die Terme höherer Ordnung (H.O.T.) werden üblicherweise ignoriert, da diese insignifikant klein sind. Setzt man nun Gleichung 2.12 in Gleichung 2.13 ein, so ergibt sich:

$$\frac{\delta I}{\delta X} \delta x + \frac{\delta I}{\delta Y} \delta y + \frac{\delta I}{\delta t} \delta t = 0 \quad (2.14a)$$

Hierbei beschreiben δx und δy die Bewegung des Pixels über die Zeit, daher kann dies auch folgendermaßen geschrieben werden:

$$\frac{\delta I}{\delta X} \frac{\delta x}{\delta t} + \frac{\delta I}{\delta Y} \frac{\delta y}{\delta t} + \frac{\delta I}{\delta t} = 0 \quad (2.15a)$$

$$\frac{\delta I}{\delta X} v_x + \frac{\delta I}{\delta Y} v_y + \frac{\delta I}{\delta t} = 0 \quad (2.15b)$$

In obiger Gleichung beschreibt v_x und v_y den optischen Fluss des Pixels x, y . Nun kann Gleichung 2.15b kompakter umgeschrieben werden indem man folgendes definiert:

$$\frac{\delta I}{\delta X} = I_x, \frac{\delta I}{\delta Y} = I_y, \frac{\delta I}{\delta t} = I_t$$

Daraus ergibt sich die nachfolgende Regel:

$$(I_x, I_y)(v_x, v_y) = -I_t \quad (2.17a)$$

Ausgehend von Gleichung 2.17 kann man nun ein Gleichungssystem aufstellen, mit dessen Hilfe der gesuchte optische Fluss berechnet werden kann.

$$A = \begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \end{bmatrix}, v = \begin{bmatrix} v_x \\ v_y \end{bmatrix}, b = \begin{bmatrix} -I_t(p_1) \\ -I_t(p_2) \\ \vdots \end{bmatrix} \quad (2.18a)$$

$$Av = b \quad (2.18b)$$

Die Lucas-Kanade-Methode löst diese Gleichung mit Hilfe von Least-Square. Hierbei wird Pixel $p1$ zum Zeitpunkt t und der korrespondierende Pixel zum Zeitpunkt $t2$ ausgemacht. Diese können beispielsweise durch den Shi-Tomasi- oder Harris-Kantendetektor ermittelt werden. Unter der Annahme, dass sich die Nachbarpixel um Pixel $p1$ nicht verändern, hat man genügend Gleichungen zur Verfügung, um das Gleichungssystem zu lösen. Das Problem ist, dass die Matrix A aus linear unabhängigen Gleichungen bestehen muss, da sonst gilt $\det(A) = 0$. Dies ist beispielsweise der Fall, wenn die Region um den betrachteten Pixel homogen ist.

Eine weitere Methode ist die von Gunnar Farnebäck welche in seiner Publikation "Two-Frame Motion Estimation Based on Polynomial Expansion" [5] beschrieben wird. Die Lukas Kanade Methode findet das Bewegungsfeld nur an bestimmten Punkten (Beispielsweise an Eckpunkten). Die Methode von Gunnar Färnebeck ist nicht auf die Detektionsalgorithmen von Harris oder Shi-Tomasi angewiesen und ist somit auch nicht auf deren Limitation beschränkt. Die Idee hierbei ist es, eine Nachbarschaft durch ein Polynom zu repräsentieren. Dies bezeichnet Farnebäck in seiner Dissertation "Polynomial Expansion for Orientation and Motion Estimation" [4] als "polynomial expansion". Die grobe Idee ist es, ein Signal oder eine Nachbarschaft eines Pixels durch Gewichtungen zu repräsentieren,

welche diese beschreibt. Dadurch kann diese Nachbarschaft in darauffolgenden Aufnahmen wieder gefunden werden. Hieraus kann der optische Fluss berechnet werden. Dazu werden zwei quadratisches Polynome f_1 und f_2 betrachtet.

$$f_1(x) = x^T A_1 x + b_1^T x + c_1$$

Das Polynom f_2 ergibt sich durch die Verschiebung von f_1 um d .

$$f_2(x) = f_1(x - d) = (x - d)^T A_1 (x - d) + b_1^T (x - d) + c_1 \quad (2.20a)$$

$$= x^T A_1 x + (b_1 - 2A_1 d)^T x + d^T A_1 d - b_1^T d + c_1 \quad (2.20b)$$

$$= x^T A_2 x + b_2^T x + c_2 \quad (2.20c)$$

Hieraus ergibt sich folgendes:

$$A_2 = A_1 \quad (2.21a)$$

$$b_2 = b_1 - 2A_1 d \quad (2.21b)$$

$$c_2 = d^T A_1 d - b_1^T d + c_1 \quad (2.21c)$$

Unter der Voraussetzung, dass A_1 nicht singulär ist, lässt sich mittels Gleichung 2.21c die Verschiebung d berechnen.

$$2A_1 d = -(b_2 - b_1) \quad (2.22a)$$

$$d = -\frac{1}{2} A_1^{-1} (b_2 - b_1) \quad (2.22b)$$

In der Publikation werden noch weitere Aspekte beschrieben, welche zu einer robusteren Methode führen. Die Grundidee, welche durch Gleichung 2.22b gegeben wird, bleibt jedoch dieselbe [5].

3 Stand der Forschung

Dieses Kapitel betrachtet die Schwarmzustände, welche in Fischschwärmenden nachgewiesen wurden. Daraufhin folgt die Definition der Modelle, mit deren Hilfe die Zustände abgebildet werden sollen.

3.1 Zustände von Schwärmenden

Viele Schwärme zeigen koordiniertes Verhalten, das durch das Zusammenspiel der einzelnen Individuen zustande kommt. Dies lässt sich auf das autonome Verhalten der einzelnen Agenten zurückführen. Schwärme, dessen Agenten den Regel von Reynolds (vgl. 2.1.1) folgen zeigen drei unterschiedliche Zustände. Diese werden in dem Papter "Collective States, Multistability and Transitional Behavior in Schooling Fish" als *swarm*, *polarized* und *milling* bezeichnet [20] .

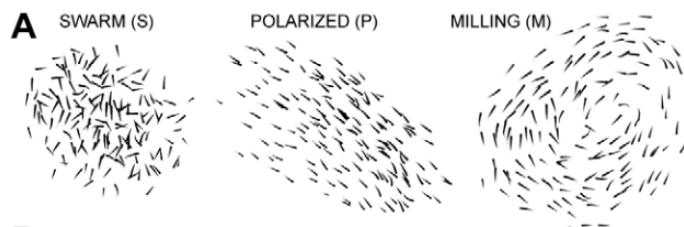


Abbildung 3.1: Zustände von Fischschwärmenden , entnommen aus [20]

Swarm bezeichnet globale und lokale Unstrukturiertheit. Hierbei ist die Ausrichtung der Agenten scheinbar zufällig. Dies ist in Abbildung 3.1 links zu sehen. Hierbei findet zum größten Teil keine Bewegung des Schwarms als Ganzes statt.

Polarized wird der Zustand, der im mittleren Bild zu sehen ist, genannt. In diesem Zustand sind die Agenten gleich ausgerichtet. Der Schwarm bewegt sich dadurch als Ganzes in eine Richtung.

Milling ist der Zustand, der im rechten Bild zusehen ist. Die Agenten kreisen um einen Punkt. Hier findet ebenfalls eine Bewegung des Schwarms statt. Der Schwarm bewegt sich hier lokal an einer Stelle.

Ist der Schwarm im *Polarized* Zustand, so ist die Richtung der Agenten größtenteils gleich. Mittelt man über die normierten Richtungsvektoren u_i der Agenten, gibt die Länge des hieraus resultierenden Vektors Aufschluss auf das Vorhandenseins dieses Zustands.

Der Zustand *Milling* kann auf ähnliche Weise ermittelt werden. Hierzu wird der Vektor r_i berechnet, der von Agent i in Richtung Mittelpunkt des Schwarms zeigt. Vergleicht man nun die Vektoren r_i und u_i so kann festgestellt werden,

ob der Schwarm rotiert. Dies ist der Fall, wenn r_i und u_i im rechten Winkel zueinanderstehen.

$$O_p = \frac{1}{N} \left| \sum_{i=1}^N u_i \right| \quad (3.1a)$$

$$O_r = \frac{1}{N} \left| \sum_{i=1}^N u_i \times r_i \right| \quad (3.1b)$$

In obenstehender Gleichung ist die Berechnungsvorschrift für den *Polarized*-Zustand (3.1a) und dem *Milling*-zustand (3.1b) zu sehen.

Eine wichtige Erkenntnis ist, dass die Zustände koexistieren können. Weiter werden die Zustände Anhand der Werte von O_p und O_r definiert.

- Der polarisierte Zustand ist erreicht wenn: $O_p > 0.65$ und $O_r < 0.35$
- Der Rotationszustand ist erreicht wenn: $O_r > 0.65$ und $O_p < 0.35$
- Der Schwarmzustand ist erreicht wenn: $O_p < 0.35$ und $O_r < 0.35$

Diese Schranken werden im weiteren Verlauf als Maßgebend für das Vorhandensein eines Zustandes sein.

3.2 Modelle

Die Modellbeschreibung, wie sie in Abschnitt 2.1.1 beschrieben wurden, definieren noch kein explizites Modell. In diesem Teil des Kapitels werden zwei Modelle vorgestellt. Auf das erste Modell trifft man des öfteren in der gängigen Literatur. Teilweise auch in abgewandelter Form. Die hier gewählte Form wird verwendet, da es in Zusammenhang mit den drei vorgestellten Zuständen steht. Das zweite Modell ist eines, welches aus eigenen Überlegungen entstand. Hierbei wird eine eigene Interpretation der Schwarmzustände in Einklang mit einem Modell gebracht.

3.2.1 Metrisches Modell

Wie bereits angesprochen, fundieren die agentenbasierten Modelle auf den drei Grundannahmen des Abstandhaltens, Angleichens und Zusammenbleibens. Der Versuch, die typischen Eigenschaften eines Schwarms zu simulieren, wurde von in dem Paper "Collective memory and spatial sorting in animal groups" präsentiert [2]. Hierbei wird auf ein Modell gesetzt, das den metrischen Ansatz verfolgt.

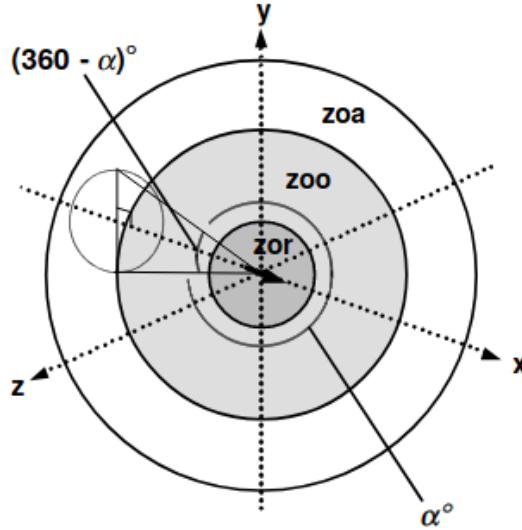


Abbildung 3.2: Zonen im metrischen Modell eines Agenten nach [2], Abbildung entnommen aus selbigem Paper

Der Wahrnehmungsbereich eines Agenten besteht aus drei Zonen. Die innerste Zone des Models ist die Abstandhaltenzone. In Abbildung 3.2 **zor** (zone of repulsion) bezeichnet. Diese Zone hat die höchste Priorität. Befinden sich Agenten in dieser Zone, so haben andere Zonen keinen Einfluss auf das Verhalten des Agenten. Dies ist laut Majolo und Huang [10] ein häufig beobachtetes Phänomen im Tierreich. Die nachfolgende Gleichung legt das Verhalten des Agenten für diesen Zonenbereich dar.

$$d_r(t + \tau) = - \sum_{j \neq i}^{n_r} \frac{r_{ij}(t)}{|r_{ij}(t)|} \quad (3.2a)$$

Finden sich Agenten in dieser Zone, so bewegt sich der Agent in Richtung $d_i(t + \tau) = d_r(t + \tau)$. Der Vektor $r_{ij}(t)$ ist der Vektor, der zum Nachbaragenten j zeigt. Gleichung 3.2a beschreibt hier das sich Entfernen des Agenten i von dem lokalen Massezentrum welches durch die Agenten j entsteht. Für den Fall, dass $n_r = 0$ ist, kommen die Zonen **zoo** (zone of orientation) und **zoa** (zone of attraction) zu tragen.

Der Agent i gleicht seine Orientierung den Orientierungen der Agenten j , die sich in der **zoo** befinden, an. Dies lässt sich durch folgende Vorschrift berechnen:

$$d_o(t + \tau) = \sum_{j=1}^{n_o} \frac{v_j(t)}{|v_j(t)|} \quad (3.3a)$$

Der Vektor $v_j(t)$ ist der Richtungsvektor des Agenten j . Er wird durch die Position c_j des Agenten zum Zeitpunkt t und $t - 1$ berechnet. Daher ist dies der Richtungsvektor, der in dem vorherigen Zeitschritt berechnet wurde.

Die zone of attraction ist die Zone, die das Zusammenhalten sicherstellen soll. Das Verhalten wird durch folgende Gleichung dargestellt.

$$d_a(t + \tau) = \sum_{j \neq i}^{nr} \frac{r_{ij}(t)}{|r_{ij}(t)|} \quad (3.4a)$$

Der Richtungsvektor, der sich aus den Agenten innerhalb der **zoa** ergibt wird ebenfalls durch die Richtungsvektoren, die von Agent i zum Nachbarn j zeigen, berechnet. Dies beschreibt die Bewegung in Richtung des lokalen Massezentrums, welches durch die Agenten j in der **zoa**-Zone entsteht. Die **zor** und **zoa** können als Gegenspieler gesehen werden. Erstere treibt die Agenten auseinander, zweitere hält die Agenten zusammen. Die Gleichungen 3.4a und 3.2a der jeweiligen Zonen unterscheiden sich daher nur durch ein Vorzeichen.

Zu beachten ist, dass die drei verschiedenen Zonen nicht überlappend sind. Dadurch ist ein beliebiger Agent eindeutig einer Zone zuzuordnen, wenn er im Sichtbereich des betrachteten Agenten ist. Durch das Aufsummieren der normierten Richtungsvektoren wird in allen Fällen die Distanz zwischen den Agenten vernachlässigt. Die hieraus resultierenden Richtungsvektoren sind jedoch nicht normiert. Dadurch vergrößert sich die Distanz, die zurückgelegt wird, wenn mehrere Agenten in den jeweiligen Zonen liegen.

Für den Fall, dass nur eine Zone besetzt ist, ergibt sich der endgültige Richtungsvektor durch die Berechnungsvorschrift der betroffenen Zone. Befinden sich hingegen nur Agenten in den Zonen **roa** und **roo**, so berechnet sich der entgültige Richtungsvektor durch $d_i(t + \tau) = \frac{1}{2}[d_o(t + \tau) + d_a(t + \tau)]$. Für den Fall, dass keine der Zonen besetzt ist, wird die vorherige Richtung beibehalten, somit ist $d_i(t + \tau) = v_i(t)$.

Jeder Agent i besitzt zudem einen blinden Kegel der hinter dem Agenten liegt. Dieser ist in Abbildung 3.2 zu sehen. Der Winkel des Kegels wird durch $360 - \alpha$ berechnet, wobei α als Wahrnehmungsbereich (*field of perception*) bezeichnet wird. Agenten in dieser Zone werden für die Berechnung des Richtungsvektors $d_i(t + \tau)$ nicht einbezogen. Ist der Winkel $\alpha = 360$ so können Individuen mit allen Agenten innerhalb der jeweiligen Zone interagieren.

Um eine zu starke Richtungsänderung zu begrenzen wird der Winkel $\theta\tau$ definiert. Ist der Winkel zwischen dem Vorherigen Richtungsvektor $v_i(t)$ und $d_i(t + \tau)$ größer als $\theta\tau$, so steuert Agent i maximal um $\theta\tau$ in Richtung $d_i(t + \tau)$. Andernfalls wird die Richtung $d_i(t + \tau)$ des Agenten nicht beschränkt.

3.2.2 Eigenes Modell

Viele Modelle in der gängigen Literatur sind metrische Modelle. Im Kontrast hierzu stehen Modelle, die topologisch aufgebaut sind. Wie in 2.1.1 besprochen

teilen sich die drei Zonen hierbei auf die n -nächsten Nachbarn auf. Ballerini u. a. [1] bezieht sich in seiner Studie auf Stare. Es wird vermutet, dass sich dies auch auf Fische übertragen lässt, da deren Sichtfeld ähnlich dem Sichtfeld von Fischen ist. Nichtsdestotrotz sind topologische Modelle in Angriffs situationen zu bevorzugen, was nicht Teil dieser Arbeit sein wird.

Hier wird ein Modell vorgestellt, das im Laufe dieser Arbeit aus eigenen Überlegungen zustande kam. Hierbei werden die drei Grundprinzipien stets eingehalten. Dieses Modell ist kein metrisches Modell und kein rein topologisches, auch wenn Aspekte beider Modell mit einbezogen werden.

Der wichtigste Faktor ist, wie bereits erwähnt, das Abstandthalten. Eigenen Überlegungen nach sollte es genügen, wenn jeder Agent i nur dem nächstgelegenen Agenten ausweicht. zieht man mehrere Agenten in Betracht und vernachlässigt deren Distanz zum betrachteten Individuum i , wie es im vorherigen Modellansatz der Fall ist, so wird der Masseschwerpunkt von dem sich der Agent weg bewegt, von weiter entfernten Individuen genau so stark bestimmt wie von näheren. Jedoch ist es dringender, sich dem Agenten, der sich in unmittelbarer Nähe befindet, auszuweichen, als weiter entfernten Agenten. Es wird daher pro Agent i der Agent j ermittelt, der nach der euklidischen Distanz am nächsten liegt. Hieraus errechnet sich der Richtungsvektor U_i des Agenten i folgendermaßen:

$$U_r(t + \tau) = -\frac{r_{ij}(t)}{|r_{ij}(t)|} \cdot f_1(|r_{ij}(t)|) \quad (3.5a)$$

Die Funktion f_1 in Gleichung 3.5a soll den normierten Richtungsvektor skalieren. Hierbei sollen weit entfernte Agenten weniger starken Einfluss auf die Richtung haben als nähere.

Die Orientierung des betrachteten Agenten wird wie im Modell zuvor berechnet. Hierbei werden allerdings alle anderen Agenten j mit einbezogen (ausgenommen Agent i).

$$U_o(t + \tau) = -\frac{v_j(t)}{|v_j(t)|} \cdot f_2(|r_j(t)|) \quad (3.6a)$$

Die Skalierungsfunktion f_2 sorgt dafür, dass zu nahe sowie zu weite entfernte Agenten einen kleinen bis keinen Einfluss auf die Orientierung des Agenten haben. In diesem Sinne ist dies auch eine Zone, allerdings ist dies hier keine binäre Entscheidung wie zuvor. Selbiges gilt für die nächste Zone.

$$U_a(t + \tau) = \frac{r_j(t) - com(t)}{|r_j(t) - com(t)|} \cdot f_3(|r_j(t) - com(t)|) \quad (3.7a)$$

$$com(t + \tau) = \frac{1}{N} \sum_i^N c_i(t) \quad (3.7b)$$

Hier wird der normierte Richtungsvektor in Richtung des Massezentrums com , welches durch alle Agenten erzeugt wird berechnet und skaliert. Die Skalierungsfunktion f_3 minimiert hierbei den Einfluss des Massezentrums je näher sich Agent i an diesem befindet. Hierdurch ist der Zusammenhalt gewährleistet und gleichermaßen wird vermieden, dass sich Agenten zu weit vom Schwarm entfernen.

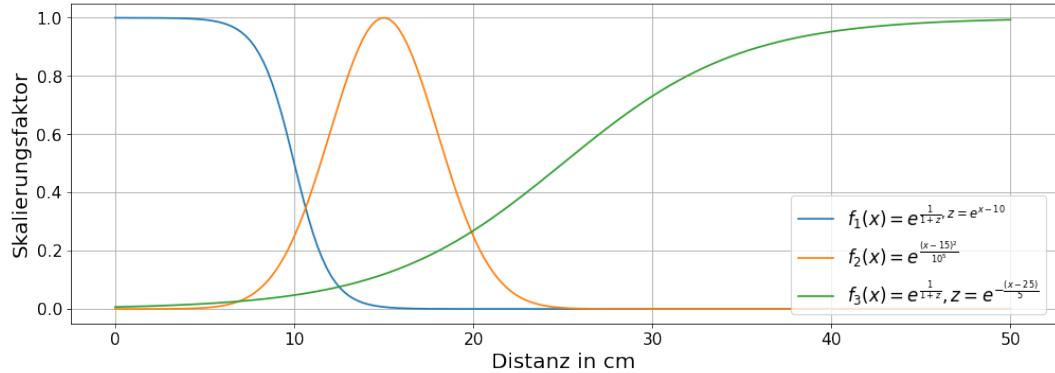


Abbildung 3.3: Skalierungsfunktionen für die verschiedenen Richtungsvektoren

In Abbildung 3.3 sind die gewählten Abstandsfunktion zu sehen. Die Faktoren wurden experimentell mit Rücksicht auf den verwendeten Datensatz bestimmt. Ziel ist es drei unterschiedliche Zonen zu erzeugen, durch die die Richtungsvektoren aus den Gleichungen 3.5a, 3.6a und 3.7a in Bezug auf die Distanz des dazugehörigen Agenten gewichtet.

Der Richtungsvektor des Agenten i setzt sich nun durch die Linearkombination aus den drei beschriebenen Vektoren zusammen:

$$d_i(t + \tau) = [\alpha_1 U_a(t + \tau) + \alpha_2 U_a(t + \tau) + \alpha_3 U_a(t + \tau)] \quad (3.8a)$$

$$U_i(t + \tau) = \gamma \frac{d_i(t + \tau)}{|d_i(t + \tau)|} \quad (3.8b)$$

Die Variablen α_i in Gleichung 3.8a gewichten den Einfluss der Zonen. Im vorherigen Modell wird das Verhalten über das justieren der Zonengröße gesteuert. Hier sind die Zonen nicht veränderbar, einzig durch die Parameter α_i ist es möglich das Verhalten zu ändern. Der Parameter γ aus Gleichung 3.8b streckt oder staucht den resultierenden Richtungsvektor und kann somit als zurückgelegter Weg betrachtet werden.

Reynolds erwähnt in seiner ursprünglichen Publikation keinen blinden Kegel sowie keine Richtungsbeschränkung. Für dieses Modell wird dies ebenfalls der Fall sein. Das hat zum einen Einfluss auf die Performance und zum anderen gibt es den Agenten einen größeren Freiraum, sich zu bewegen. Nachteil ist hierbei, dass es zu unnatürlichen Bewegungsmustern kommen kann. Beispielsweise könnte ein Agent in einem nächsten Zeitschritt die entgegengesetzte Richtung einnehmen.

Dies ist allerdings nur der Fall, wenn nur der Agent in unmittelbarer Laufrichtung Einfluss nimmt.

Ein wichtiger Unterschied zwischen den beiden Modellen ist an dieser Stelle anzumerken. Modell aus dem Abschnitt 3.2.1 kennt das Massezentrum der Agenten nicht. Das in diesem Abschnitt vorgestellte Modell jedoch kennt das Massezentrum. Zu erkunden ob diese Annahme realistisch ist, soll allerdings nicht Teil dieser Thesis sein.

4 Experimente

4.1 Tracking

Für das Approximieren von Parametern anhand von realen Daten wird ein geeigneter Datensatz benötigt. Insbesondere werden die Trajektorien der Fische benötigt. Fische haben hierbei den Vorteil, dass sie pseudo-dreidimensional sind. Werden die Schwärme in einem relativ niedrigen Becken gehalten, so kann sich auf deren X und Y Komponente beschränkt werden. Natürlich existiert hierbei die Z-Komponente, welche aber durch das flache Becken keine große Ausprägung erfährt. Eine Konsequenz einer existierenden Z-Komponente ist, dass Fische sich überlagern können und dadurch die Problematik entsteht, diese im Prozess des Trackens trennen zu müssen. Ein weiterer Vorteil ist, dass das Tracken von Schwärmen, deren Z-Komponente nicht vernachlässigbar ist, ein wesentlich komplexeres Unterfangen ist. Betrachtet man einen Vogelschwarm, so muss dieser mit zwei kalibrierten Kameras aufgenommen werden. Hieraus resultieren Aufnahmen des Vogelschwarms, welche auf zwei Ebenen aufgeteilt sind (Beispielsweise die XY und XZ-Ebene). Um daraus die dreidimensionalen Positionen der Vögel zu extrahieren, müssen diese Ebenen registriert werden, was Material für eine eigenständige Masterthesis liefert. Des Weiteren besteht das Problem, dass sich der Schwarm als Ganzes im Raum bewegt, die Kameras jedoch stationär sind. Eventuell fliegen einzelne Individuen weit genug weg, um von der Kamera nicht erfasst werden zu können. Aus den genannten Gründen wird sich im weiteren Verlauf auf die Trajektorien von Fischen beschränkt.

Es wird ein Datensatz, welcher aus Aufnahmen von Fischschwärmen der Spezies *juvenile zebrafish* besteht, verwendet. Die Größe eines ausgewachsenen Zebrafisches liegt zwischen 10.0 und 19.5 mm. Die Videos der Schwärme wurden mit einer Auflösung von 3712 x 3740 Pixeln aufgenommen. Die Bildrate beträgt 32 Bilder Pro Sekunde. Die verwendete Kamera ist eine *Emergent Vision HT-20000M* welche mittig 70mm über dem Fischbecken montiert ist. Das kreisförmige Fischbecken ist gefüllt mit 28 Grad warmen Wasser, welches der benötigten Wassertemperatur der Zebrafische entspricht. Die Wassertiefe beträgt 2,5 cm und der Durchmesser des Beckens beträgt 70 cm. Der Aufbau des Experiments ist in nachfolgender Abbildung zu sehen.

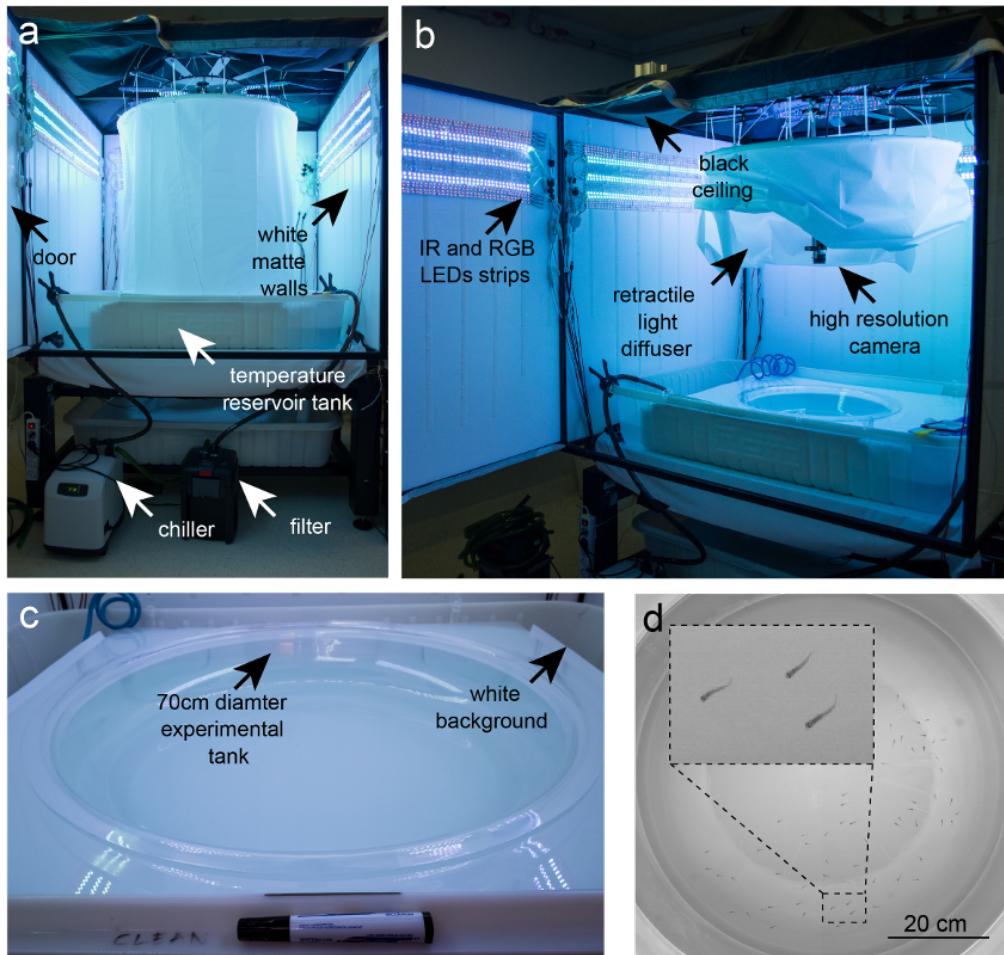


Abbildung 4.1: Versuchsaufbau für die Aufnahme der Fischschwärme entnommen aus <https://idtrackerai.readthedocs.io/en/latest/setups.html> (besucht am 24.11.2021)

Die Dimension des Versuchsaufbaus und die Auflösung der Kamera ermöglicht das Umrechnen der Pixel in Zentimeter. Durch die Bildrate und die Umrechnung der Pixel ist es möglich, die zurückgelegte Distanz pro Zeiteinheit zu bestimmen. Diese Tatsache wird im weiteren Verlauf genutzt, um einen Bezug zu realen Größen zu erhalten.

4.1.1 Eigene Trackingmethode

Für das Schätzen der Bewegung von Individuen wird der optische Fluss von Farnebäck verwendet. Die Idee ist es, Individuen zwischen zwei Frames auszumachen und diese anschließend einander zuzuordnen. Hierzu wird per optischen Fluss das Bewegungsfeld zwischen zwei Frames geschätzt. Daraufhin kann ein Individuum aus dem ersten Frame zum Zeitpunkt t durch das Bewegungsfeld an seine Position zum nächsten Zeitpunkt bewegt werden. Die hieraus resultierende

Position sollte nahe an der realen Position des Individuums zum Zeitpunkt $t + 1$ sein. Es wird also das nächstgelegene Individuum zugewiesen. Um keine doppelte Belegung zu erzeugen, steht dieses Individuum daraufhin nicht mehr zur Auswahl.

Um zu untersuchen, wie gut diese Methode funktioniert, wird anhand einer Simulation das Tracken von künstlich erzeugten Daten durchgeführt. Von Interesse ist insbesondere die Genauigkeit des Trackens in Bezug auf die Distanz, die die Agenten zurücklegen. Erwartet wird, dass ab einer bestimmten Distanz, die die Agenten pro Zeiteinheit zurücklegen, die Genauigkeit des Trackens abnimmt. Zudem ist von Interesse, welche Anzahl von Agenten diese Methode bewältigen kann.

Es sind zwei Szenarien vorgesehen, die hier betrachtet werden. In Szenario eins bewegen sich Agenten horizontal über eine Szene. Dies entspricht dem polarisierten Zustand der Fische, welcher in den Aufnahmen der realen Daten zu sehen ist. Das zweite Szenario stellt den rotierenden Zustand dar. Hierbei kreisen die Agenten um einen Mittelpunkt.

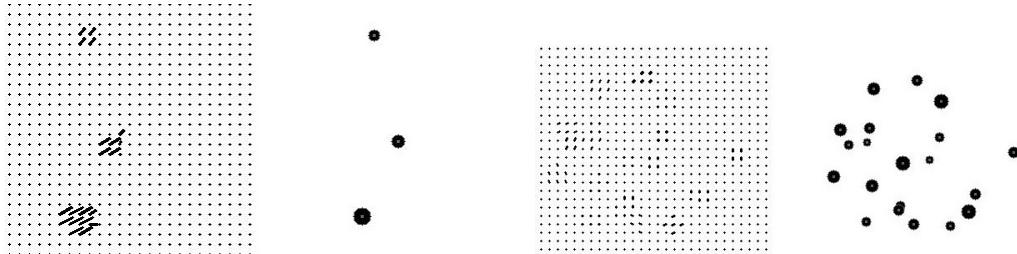


Abbildung 4.2: Tracking anhand einer Simulation, horizontaler Verlauf links, rotierender Verlauf rechts

Der linke Teil der jeweiligen Abbildung stellt den optischen Fluss dar. Wie zu sehen ist, verläuft die Richtung nicht exakt horizontal. Die Richtungsvektoren, welche die Agenten bewegt, wird in X- und Y-Richtung per Normalverteilung ($X \sim \mathcal{N}(0, 1)$) abgelenkt. Somit bewegen sich alle Agenten etwas unterschiedlicher, im Mittelwert jedoch mit gleichbleibender Geschwindigkeit. Für die Simulation sind die Positionen der Agenten zu jeder Zeit bekannt. Das bedeutet insbesondere, dass Agenten, die sich berühren, nicht zu einer Position zusammengefasst werden, wie es der Fall ist, wenn die Positionen aus den Aufnahmen geschätzt werden müssen. Hierbei stellen Individuen, die sich berühren, ein gesondertes Problem dar, welches in diesem Versuch vernachlässigt wird.

Das Experiment wird für 10,30 und 60 Fische durchgeführt, wobei die Anzahl der Simulationsschritte 300 beträgt.

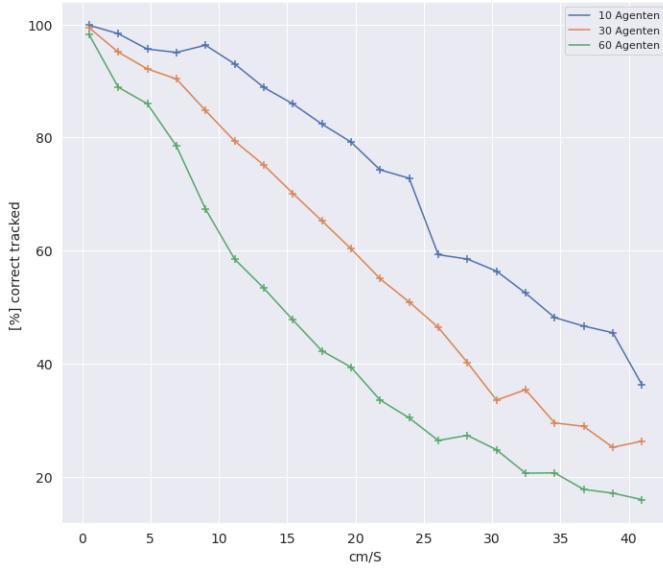


Abbildung 4.3: Ergebnisse der Trackingmethode, polarisierte Bewegung

Das Diagramm in Abbildung 4.3 zeigt das Resultat dieses Experimentes. Die x-Achse beschreibt die zurückgelegte Distanz pro Sekunde. Die y-Achse die korrekte Rate der Zuweisung der Agenten. Für die polarisierte Bewegung ist zu erkennen, dass die Anzahl der Agenten eine signifikante Rolle für die Genauigkeit des Trackens spielt. Für 10 Fische befindet sich die Genauigkeit bei $6\text{cm}/\text{S}$ noch bei über 95%, bei 60 Fischen hingegen bei unter 80 %. Ebenfalls ist zu sehen, dass die Genauigkeit mit steigender Geschwindigkeit abnimmt.

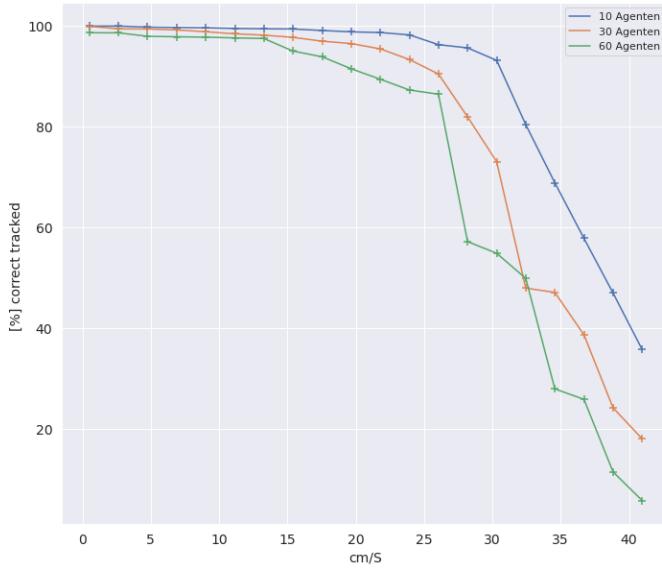


Abbildung 4.4: Ergebnisse der Trackingmethode, rotierende Bewegung

Für die Millingformation funktioniert das Tracken wesentlich besser als für

die polarisierte Formation. Hier beginnt der Abfall der Genauigkeit erst ab ca 25 cm/S . Dies kann durch die bessere Verteilung der Agenten in den einzelnen Szenen erklärt werden. Während die Agenten in der polarisierten vorm hauptsächlich in direkter Nähe zueinander positioniert sind, befinden sich die Agenten in der Millingposition auf einer größeren Fläche. Hierdurch sind die Agenten nicht so nahe beieinander positioniert. Daraus resultiert eine bessere Zuordnungsrate.

Um nun zu ergründen, wie gut diese Methode auf den Realdaten funktioniert, wird mit optischem Fluss die Bewegungsvektoren der Fische zwischen einigen Paaren an Frames berechnet. Die Länge der Vektoren ergibt hierbei in etwa die zurückgelegte Strecke der Fische pro Zeiteinheit. Im Mittelwert schwimmen die Schwärme für 10 Fische ca 4cm/S . Die maximale Geschwindigkeit überschreitet 10cm/S nicht, wodurch sich eine Genauigkeit beim Tracking von über 95% ergibt. Für 60 Fische ergibt sich eine mittlere Geschwindigkeit von 5cm/S und eine maximale Geschwindigkeit von über 8cm/S . Das bedeutet, dass die Genauigkeit im Mittelwert bei knapp unter 90% liegt und für den maximalen Wert bei ca. 70%. Für eine geringe Anzahl an Fischen scheint diese Methode ausreichend gut zu funktionieren. Will man nun Parameter approximieren, so ist es essenziell, dass die Positionen und Trajektorien der Fische so genau wie möglich sind. Jeder Fehler mindert hierbei die Verlässlichkeit der Approximation.

4.1.2 Tracking via Neural Network

Die Trackingmethode die von Romero-Ferrero u. a. [17] verwendet wird, setzt auf künstliche neuronale Netzwerke. Die Nachfolgende Abbildung fasst die Tracking Schritte zusammen.

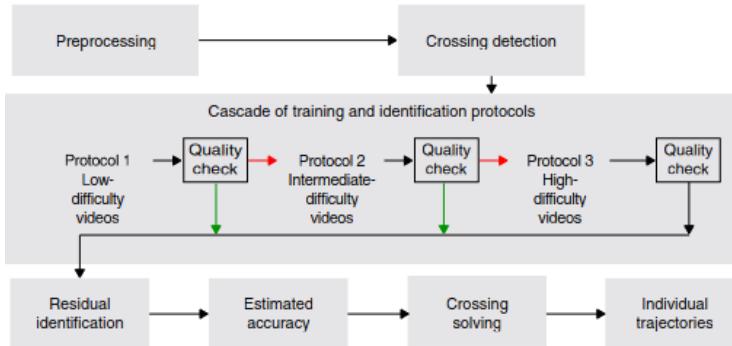


Abbildung 4.5: Ablaufdiagramm des Trackingprozesses, entnommen aus Romero-Ferrero u. a. [17]

In einem Vorverarbeitungsschritt werden alle Individuen pro Bild extrahiert. Hierbei wird das Bild in Binärformat gebracht. Der Hintergrund wird auf null gesetzt, der Vordergrund (die Fische) auf eins. Um herauszubekommen, welche Pixel als Hintergrund betrachtet werden, wird der Mittelwert von einigen Bildern der betrachteten Aufnahme berechnet. Subtrahiert man nun diesen Mittelwert

von einem Bild der Aufnahme, so negiert sich der Hintergrund, während sich der Vordergrund abhebt (dies funktioniert nur, wenn der Hintergrund statisch ist). Nun wird sequenzielles Labeling verwendet, um die Positionen der Fische zu berechnen. Zu kleinen Regionen, welche beim Labeling extrahiert werden, werden verworfen.

Eines der Hauptprobleme des Trackings ist es, sich berührende und schneidende Individuen zu erkennen. Dieser Schritt wird von einem künstlichen neuronalen Netzwerk übernommen.

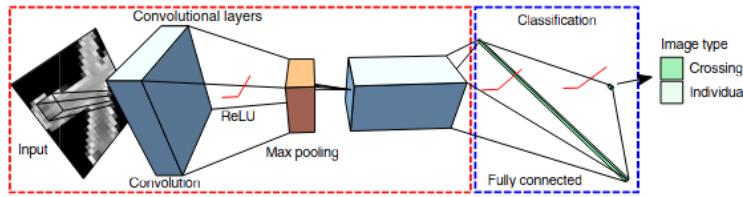


Abbildung 4.6: Netzwerkarchitekturen zur Klassifikation von sich berührenden und separaten Individuen entnommen aus Romero-Ferrero u. a. [17]

Die zuvor extrahierten Blobs werden durch das Netzwerk in sich schneidend und individuell klassifiziert. Das Netzwerk wird automatisch anhand des Videos trainiert. Dazu muss die Anzahl der Individuen im Vorfeld bekannt sein. Das Datenset für das Training wird folgendermaßen zusammengestellt. Wurden exakt so viele Individuen aus dem Bild extrahiert wie vorgegeben, so werden diese extrahierten Fische als Individuen angesehen. Andernfalls werden die extrahierten Blobs (Zusammenschluss von Pixeln) nach einem Entscheidungskriterium als sich berührend markiert. Das Kriterium ist die Abweichung der Anzahl an Pixel pro Blob. Ist diese mehr als vier Standardabweichungen vom Median entfernt, so berühren sich die Fische. Es ist hervorzuheben, dass die Identifikation von sich berührenden Fischen auch ohne künstliches neuronales Netzwerk funktioniert. Der Laufzeitvorteil eines KNNs und die erreichbare Genauigkeit sind hierbei allerdings von Vorteil.

Daraufhin übernehmen die Protokolle 1-3 (siehe Abbildung 4.5) die Aufgabe das Identifikationsnetzwerk, welches in der nachfolgenden Abbildung zu sehen ist, zu trainieren.

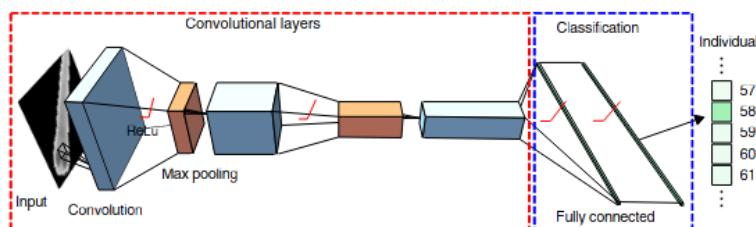


Abbildung 4.7: Netzwerkarchitekturen zur Identifikation von Individuen entnommen aus Romero-Ferrero u. a. [17]

Innerhalb Protokoll 1 werden alle Intervalle zusammengestellt (pro Fisch), in denen die Fische separat voneinander sind. Diese Sequenzen sind unterschiedlich lang, enthalten aber keine Bilder von sich berührenden Fischen. Diese Intervalle werden als "global fragment" bezeichnet. Daraufhin wird die kürzeste Distanz ermittelt, die ein Individuum innerhalb eines "global fragments" zurücklegt. Diese Fragmente werden nun verwendet, um das Identifikationsnetzwerk, das in Abbildung 4.7 zu sehen ist zu trainieren. Die Protokolle 2 und 3 treten in Kraft, wenn Protokoll 1 eine unzureichende Genauigkeit im Zuweisen von Identitäten aufweist. Diese Protokolle verfeinern das Training aus Protokoll 1 um bessere Ergebnisse zu erzielen.

Das Post-processing beginnt mit dem Zuweisen der Identitäten in Form von Nummerierung (Residual identification). Hierzu wird das finale Netzwerk aus den vorherigen Trainingsschritten verwendet. Daraufhin folgt die Bestimmung der Genauigkeit mittels Bayesian Framework. Dazu werden die innerhalb der Protokolle extrahierten globalen Fragmente verwendet. Schlussendlich werden die Individuen, welche sich berühren, via Erosion und Interpolation getrennt. Aus diesen Schritten können die Trajektorien der Fische zusammengesetzt werden.

Video	Pixels per animal (segmentation)	Pixels per animal (identification)	File size (Gb)	Tracking time	Protocol	Accuracy
8 zebrafish	331	599	9.14	0:30:51	2	99.969
	168	361	5.14	0:19:38	2	99.893
	63	193	2.29	0:19:04	2	99.890
	23	100	1.12	0:29:34	2	99.618
	9	61	0.57	0:23:42	2	96.690
	2	28	0.20	4:15:31	3	48.366
60 zebrafish	303	578	228.02	5:18:41	2	100
	163	367	128.26	5:14:47	2	99.998
	64	193	57.00	7:05:52	2	99.998
	28	104	27.92	15:18:29	3	99.989
	12	63	14.25	12:42:58	3	89.187
	3	28	5.13	1 day, 22:33:39	3	19.572

Abbildung 4.8: Auswertung der Trackinggenauigkeit entnommen aus Romero-Ferrero u. a. [17]

In Abbildung 4.8 ist die Auswertung der Trajektorien zu sehen. Hier wurden Videos von 8 und 60 Zebrafischen betrachtet. Das Tracken wurde jeweils mit unterschiedlichen Auflösungen durchgeführt. In der Protokollspalte ist zu sehen, dass für das Training mindestens Protokoll zwei zum Einsatz kam. Die Accuracy wurde hier von Hand durch das Inspizieren der Trajektorien anhand von 3000 Bildern durchgeführt. Zu sehen ist, dass die Genauigkeit bei nahezu 100% liegt. Die Trajektorien aus dieser Trackingmethode ist somit der eigenen Methode vorzuziehen und wird daher für das Approximieren der Parameter im Laufe der Thesis verwendet. Die Trajektorien für 10,60,80 und 100 Fische stehen auf der Homepage von idtracker.ai zur Verfügung.

4.1.3 Eigenschaften des Datensatzes

Durch die im vorherigen Abschnitt besprochene Trackingmethode stehen die Trajektorien in Form von x-,y-Positionen pro Individuum zur Verfügung. Das

ermöglicht Einsicht in das Verhalten der Schwärme. Im vorherigen Abschnitt wurde die Geschwindigkeit der Fische mit optischem Fluss geschätzt. Nun kann durch die Positionen die exakte Geschwindigkeit pro Fisch durch die zurückgelegte Distanz zwischen zwei Frames ermittelt werden.

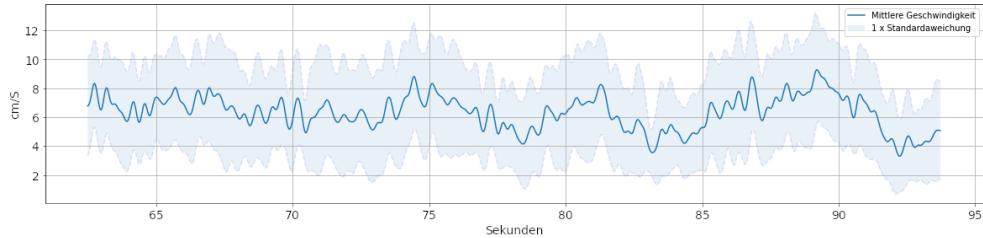


Abbildung 4.9: Mittlere Geschwindigkeit des Fischschwarms der Größe 10

Wie anhand der Abbildung 4.9 zu sehen ist schwankt die mittlere Geschwindigkeit des Schwarmes pro Zeitschritt um 7 cm/S . Die geschätzte Geschwindigkeit via optischem Fluss, liegt bei 4cm/S . Das bedeutet, dass die tatsächliche Genauigkeit mittels eigener Trackingmethode deutlich schlechter wäre als ursprünglich angenommen.

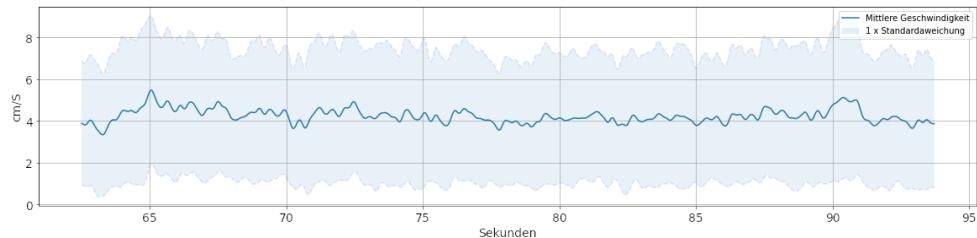


Abbildung 4.10: Mittlere Geschwindigkeit des Fischschwarms der Größe 60

Die mittlere Geschwindigkeit für 60 Fische liegt nahe an der mit optischem Fluss geschätzten Geschwindigkeit von 5cm/S . Im direkten Vergleich der beiden Diagramme kann festgestellt werden, dass die Geschwindigkeit des Schwarmes mit Zunahme der Größe abnimmt. Dies kann natürlich auch dem Umstand geschuldet sein, dass sich der Schwarm in einem begrenzten Behälter befindet und ein kleinerer Schwarm mehr Raum ausnutzen kann. Dadurch kann ein Fisch eine längere Strecke geradeaus schwimmen und somit eine höhere Geschwindigkeit erreichen. Interessanterweise zeigt ein Blick auf die Geschwindigkeit des Schwarmes mit 100 Individuen, dass dieser sich etwas schneller bewegt als der Schwarm mit 60 Fischen.

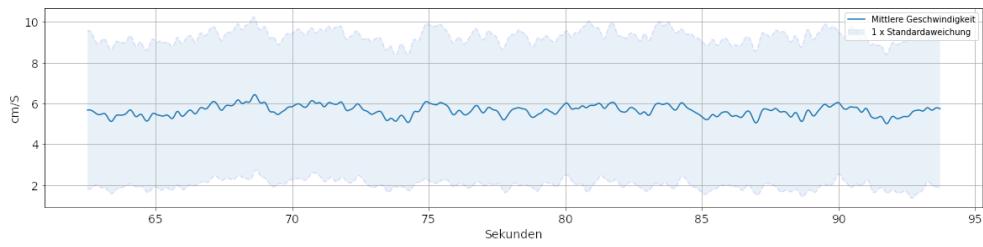


Abbildung 4.11: Mittlere Geschwindigkeit des Fischschwarms der Größe 100

Wie in Abbildung 4.11 zu sehen ist, liegt die mittlere Geschwindigkeit bei ca. 6 cm/S. Bei dem etwas kleineren Schwarm mit 60 Fischen hingegen bei 4 cm/S. Die Schwarmgröße alleine scheint die Geschwindigkeitsunterschiede nicht zu erklären. Visualisiert man den Schwarmzustand (vgl. Abschnitt 3.1), so kann man unterschiedliche Verhaltensweisen in Bezug auf die Schwarmgröße erkennen.

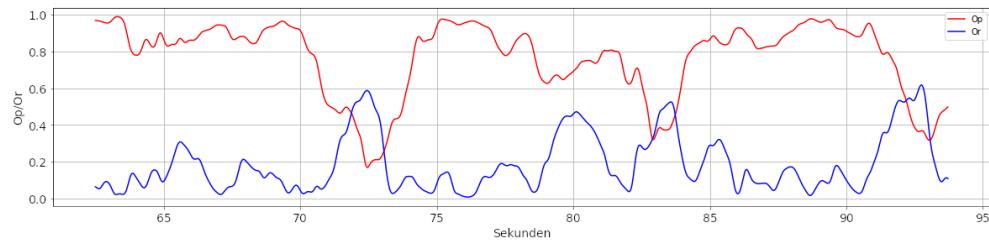


Abbildung 4.12: Zustände des Fischschwarsms der Größe 10

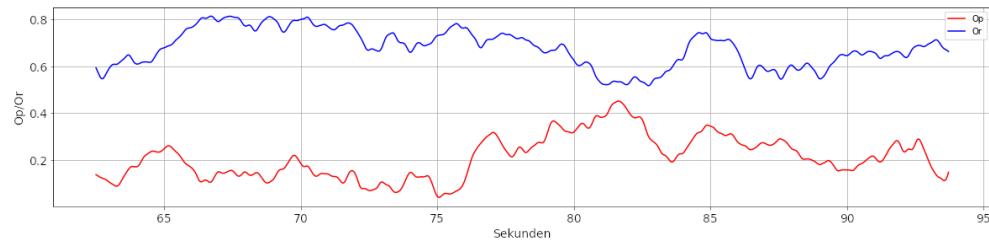


Abbildung 4.13: Zustände des Fischschwarsms der Größe 60

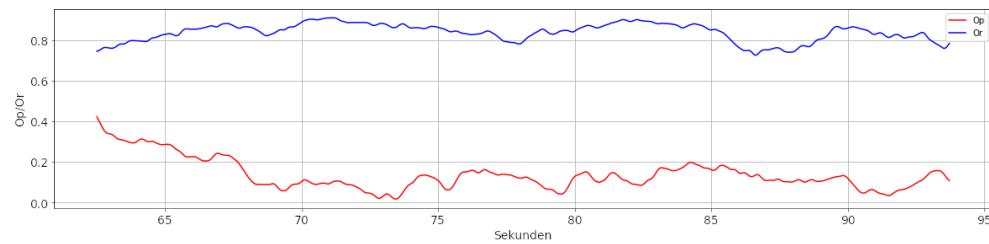


Abbildung 4.14: Zustände des Fischschwarsms der Größe 100

In den Abbildungen 4.12 - 4.14 sind die Zustände Polarisation und rotierend zu sehen. Blau ist hier der Rotationszustand und rot der polarisierte Zustand. Es ist deutlich zu erkennen, dass bei dem kleineren Schwarm mit 10 Fischen der polarisierte Zustand dominiert. Je größer der Schwarm ist, desto stärker geht er in den Rotationszustand über. Nun sieht man, dass beim Schwarm mit 60 Fischen der polarisierte Zustand noch teilweise vorhanden ist. Beim Schwarm mit 100 Fischen dominiert hingegen der Rotationszustand. Dies könnte den Geschwindigkeitsunterschied zwischen 100 und 60 Fischen, der zuvor ausgemacht wurde, erklären. Die Beobachtung ist somit, dass Schwärme, die sich in einem eindeutigen Zustand befinden, eine höhere Bewegungsgeschwindigkeit aufweisen als Schwärme, deren Zustände weniger klar definiert sind.

4.2 Approximation Anhand künstlich erzeugter Daten

In diesem Teil des Kapitels werden zu Beginn die beiden Optimierungsverfahren verglichen. Hierbei wird die Laufzeit der Algorithmen sowie die Qualität des erreichten Optimums begutachtet. Ziel in diesem Experiment ist es, die beiden Optimierungsverfahren gegenüberzustellen und herauszufinden, welcher Algorithmus in welchen Situationen zu bevorzugen ist.

Hierzu werden durch die vorgestellten Modelle künstliche Daten generiert. Initial werden den Agenten Positionen zugewiesen, die aus einer Gleichverteilung stammen. Dadurch ist die Dichte der Agenten nicht zu hoch, wie es beispielsweise bei einer Normalverteilung der Fall wäre. Das würde zur Folge haben, dass die Agenten zu Beginn konzentriert um den Mittelwert positioniert sind und dadurch zuerst den Abstand herstellen müssten. Für das Boids Modell ist in diesem Fall nur die Abstandshaltzone relevant. Jeder Agent bekommt zu Beginn einen Richtungsvektor, der aus einer Gleichverteilung gezogen wird, zugewiesen. Dieser Vektor wird hierfür normiert. Dies ist nötig, um den Agenten eine Startrichtung zu geben. Wie in Abschnitt 3.2 diskutiert, benötigen Agenten die sich in keiner Nachbarschaft befinden einen Richtungsvektor. Ohne diesen kann der Richtungsvektoren welcher sich aus der Orientierungszone ergibt, nicht errechnet werden.

4.2.1 Vergleich zwischen PSO und RMD

Das Ziel dieses Vergleiches ist zu erörtern, wie die beiden Algorithmen in einem einfachen Szenario performen. Hierzu wird eine zufällige Sequenz erzeugt, die für beide Algorithmen gleich ist.

Die Algorithmen sollen nun das Parameterset bestimmen, die diese Sequenz erzeugt. Hierbei interessiert, ob die Parameter korrekt bestimmt wurden und wie lange die Algorithmen für die Bestimmung benötigen.

Es wurden 3 Szenarien mit unterschiedlicher Schwarmgröße und gleichen Parametern erstellt. Die Schwarmgrößen sind 10,50 und 100 Agenten. Beide Algorithmen

wurden 1000 Iterationen durchlaufen. Die Lernrate für RMD ist $1e - 2$ und die Anzahl der Partikel für PSO ist 50 pro Dimension.

Als Fehlerfunktion wird die $L2$ Verlustfunktion verwendet, diese ist definiert durch:

$$L2Loss = \sum_i^n (Y_{true} - Y_{pred})^2$$

Es wird hierbei die Distanz zwischen der korrekten Agenten Position und der vorhergesagten Agenten Position bestimmt. Durch die Quadrierung werden Distanzunterschiede stärker bestraft.

Metrisches Boids Modell

Das Boids Modell muss für diesen Versuch verändert werden. Das grundlegende Problem ist, dass die Parameter nicht Teil des Graphen sind (siehe Abschnitt 2.3.1). Die Agenten werden in Folge einer binären Entscheidung den Zonen zugeordnet. Ist ein Agent innerhalb der Abstandhaltenzone, so wird er für die Berechnung mit einbezogen, anderenfalls nicht. Die Berechnung der neuen Richtung wird dadurch nur indirekt durch die Zonen bestimmt. Somit werden die Parameter nicht bis zur Fehlerfunktion durchgereicht, wodurch die Ableitung bestimmt wird.

Nun kann man auch die Distanzen der Agenten i zu dem betrachteten Agenten j mit dem Radius der jeweiligen Zone skalieren. Das führt dazu, dass, falls die Agenten i relevant sind, diese innerhalb des Einheitskreises um den Agent j liegen. Die Agenten mit dem Abstand größer als eins, werden dementsprechend ignoriert. Das hat zur Folge, dass die Radien per Skalierung in dem Ableitungsgraphen auftauchen und nach diesen abgeleitet werden kann. Der Einschlagwinkel und der Winkel der blinden Zone werden zunächst als konstant angenommen.

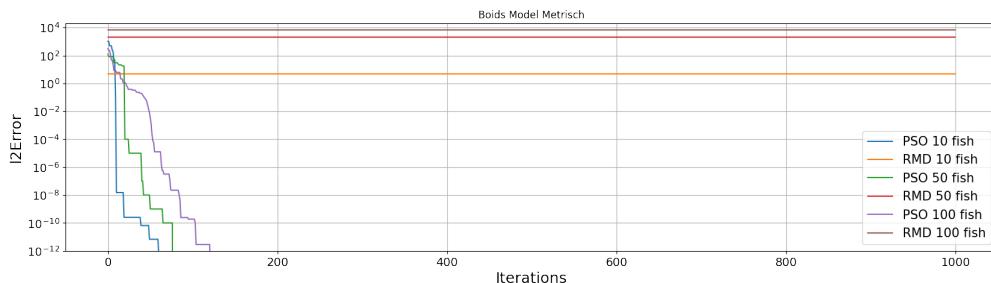


Abbildung 4.15: PSO vs. RMD anhand von Boids

Wie anhand der Abbildung 4.15 zu erkennen ist, erreicht der PSO Algorithmus im Falle des Boids Modells für alle Agentenzahlen einen minimalen Fehlerwert. Hierfür wird nicht mehr als 200 Iterationen benötigt.

Bei RMD hingegen ist keine Fehlerveränderung festzustellen. Beim Betrachten der Parameter konnte festgestellt werden, dass sich keine Veränderung einstellt.

Die Laufzeit ist hierbei nicht relevant, da sich der Fehler im Falle des RMD nicht verbessert. Somit ist hier PSO zu bevorzugen.

Eigenes Modell

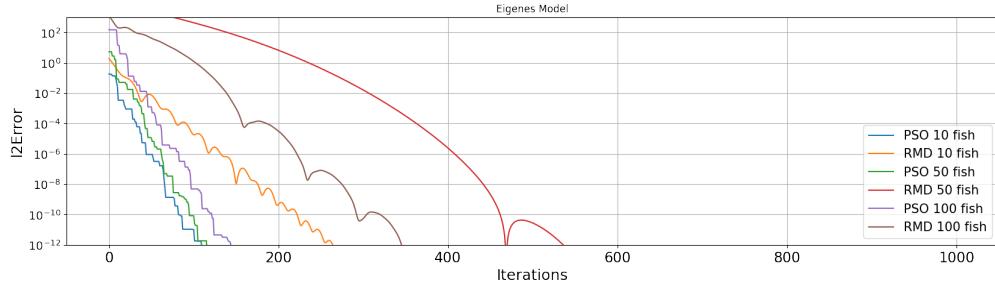


Abbildung 4.16: PSO vs. RMD Anhand des eigenen Modells

Abbildung 4.16 zeigt, dass sich für beide Verfahren eine Verbesserung des Fehlers einstellt. PSO benötigt für alle Agentenanzahlen weniger Iterationen als RMD. Für die Laufzeitbestimmung wurden die Versuche pro Anzahl der Agenten zehnmal durchgeführt. In folgender Tabelle ist der Mittelwert der jeweiligen Laufzeiten für 1000 Iterationen zu sehen.

Anzahl Agenten	Laufzeit in Sekunden PSO	Laufzeit in Sekunden RMD
10	85.5	0.6
50	262	1.3
100	598	2.6

Tabelle 4.1: Vergleich der Laufzeiten zwischen PSO und RMD

Zu erkennen ist, dass PSO in allen Szenarien schlechtere Laufzeiten liefert. Auch wenn PSO weniger Iterationen benötigt als RMD ist PSO aufgrund des großen Laufzeitunterschiedes nicht verwendbar. Aus diesem Grund wird für dieses Modell im Laufe der Arbeit der RMD Algorithmus verwendet.

4.2.2 Approximierung von konstanten Parametern

Für diesen Teil der Arbeit werden künstliche Daten mit konstanten Parametern erzeugt. Initial werden den Agenten wieder zufällige Positionen und Richtungsvektoren zugewiesen. Anhand der Daten sollen die Parameter geschätzt werden, die diese Sequenzen erzeugen.

4.2.2.1 Metrisches Boids Modell

Wie zuvor erörtert, ist für dieses Modell PSO zu verwenden. Es werden wieder 50 Partikel pro Dimension verwendet, die Anzahl der Iterationen, bis die Suche nach Parametern abgebrochen wird ist 300.

Die Simulation, auf die sich die nachfolgenden Abbildungen beziehen, sind mit 10 Agenten durchgeführt worden.

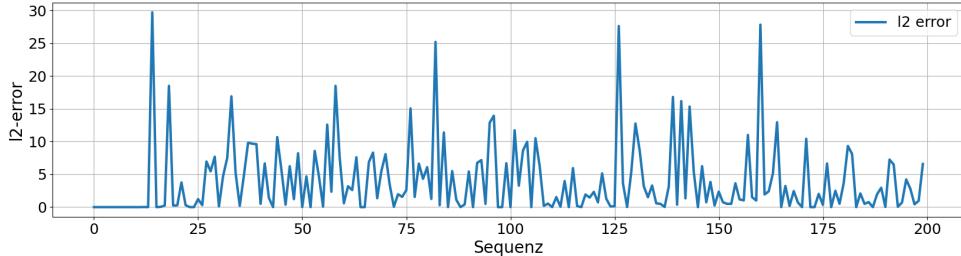


Abbildung 4.17: L2-Fehler der konstanten Parametervorhersage

Der L2-Fehler in Abbildung 4.17 ist in den ersten Sequenzen nahe an null und wird daraufhin größer. In dem vorherigen Versuch, wurden nur zwei aufeinanderfolgende Sequenzen betrachtet. Hier werden hingegen mehrere aufeinanderfolgende Sequenzen betrachtet. Hier ist es nun der Fall, dass die Vorhersage zum Zeitpunkt t , als Input für die Vorhersage zum Zeitpunkt $t + 1$ genutzt wird. Das hat zur Folge, dass die zukünftigen Sequenzen von deren Vorgängern abhängig sind. Zu erkennen ist, dass der Fehler immer wieder auf null zurückspringt.

Zudem schwankt der L2-Fehler von Sequenz zu Sequenz sehr stark. Eine Explosion des Fehlers in dem Sinne, dass dieser unweigerlich größer wird, ist hier nicht festzustellen. Wie anhand der x-Achse zu sehen ist, wurden hier Parameter für 200 Sequenzen bestimmt. Die aus der Approximation resultierenden Parameter sind in den nächsten Abbildungen zu sehen. Hierbei ist anzumerken, dass diese mittels Gaußglocke geglättet wurden, um das Rauschen der vorhergesagten Parameter zu unterdrücken. Der Parameter σ der Gaußglocke wurde für die Glättung auf den Wert 6 gesetzt.

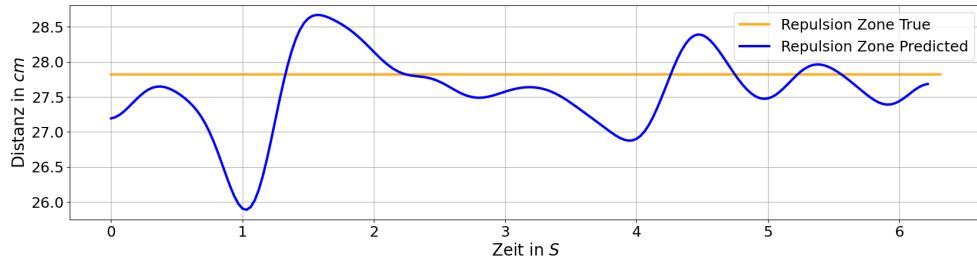


Abbildung 4.18: Vorhergesagte Abstandhalten-Zonen in Blau vs korrekte Zonen in Orange

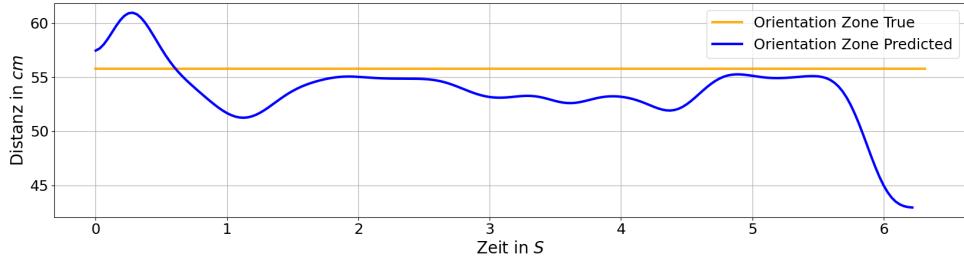


Abbildung 4.19: Vorhergesagte Orientierungs-Zonen in Blau vs korrekte Zonen in Orange

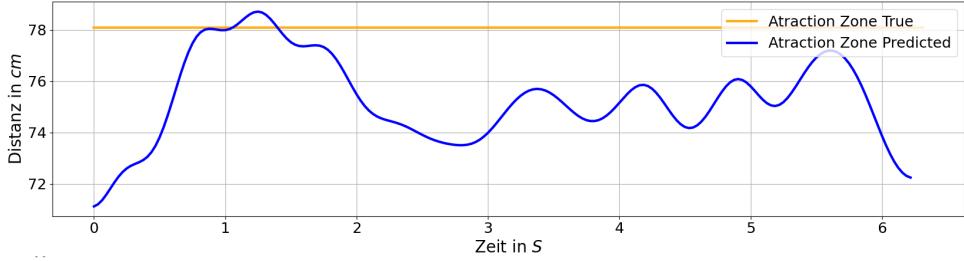


Abbildung 4.20: Vorhergesagte Zusammenhalten-Zonen in Blau vs korrekte Zonen in Orange

Die x-Achse spiegelt die Zeit wieder und die y-Achse der Radius der jeweiligen Zone in Zentimetern. In den Abbildungen 4.18-4.20 sind die konstanten Simulationsparameter für die jeweiligen Zonen zu sehen. Es ist ersichtlich, dass die vorhergesagten Parameter für die Zonen nicht konstant sind. Dies ist nicht verwunderlich, denn die Zonen des Boids Modells sind nicht sensibel gegenüber Veränderung. Ist beispielsweise ein Agent in der Orientierungszone, dann ist es nicht relevant, wie groß die Zone ist, solange nur dieser eine Agent in der Zone vorhanden ist. Selbiges gilt für die anderen Zonen.

Die Vorhersage der Winkel θ zur Begrenzung des Einschlagwinkels und α , welches den Winkel des blinden Kegels hinter dem Agenten steuert, ist in den nachfolgenden Abbildungen zu sehen.

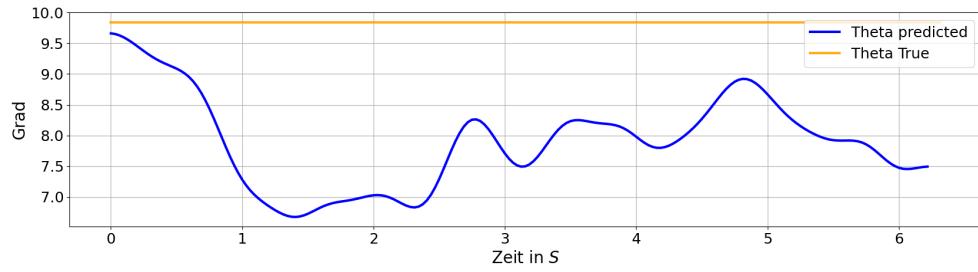


Abbildung 4.21: Vorhergesagter Einschlagswinkel in Blau vs korrekter Winkel in Orange

Der maximale Einschlagswinkel in Abbildung 4.22 wird niedriger vorhergesagt, als er tatsächlich ist. Tatsächlich sollte er bei ca. 10 Grad sein, die Vorhersage liegt im Mittel etwa bei 8 Grad.

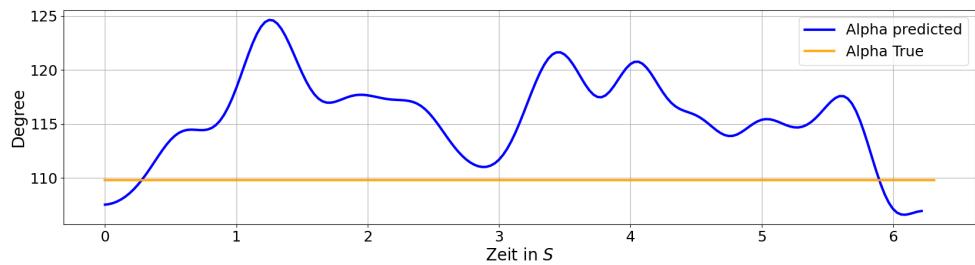


Abbildung 4.22: Vorhergesagte Winkel der blinden Zone in Blau vs korrekter Winkel in Orange

Der Winkel α der den Blindspot definiert, wird hingegen höher vorhergesagt, als er tatsächlich ist. Somit nimmt der Agent in der Vorhersage weniger Einfluss von Agenten im hinteren Bereich als Agenten, die der korrekten Simulation zugehörig sind.

Schaut man sich nun die Zustände an, so wird deutlich, dass unterschiedliche Parameter ähnliche Statusverläufe erzeugen können. Die Diagramme wurden ebenfalls mittels Gaußglocke geglättet wobei der Parameter $\sigma = 6$ gesetzt wurde. Die ungeglätteten Diagramme sind im Anhang zu sehen.

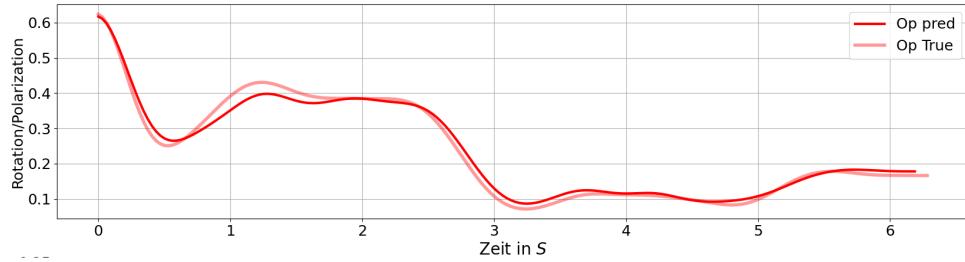


Abbildung 4.23: Polarisierungszustand der Simulation im Vergleich zur Vorhersage

In Abbildung 4.23 ist der Polarisationszustand der Simulation (transparent) sowie der Vorhersage (intransparent) zu sehen. Es ist zu erkennen, dass die Zustände nahe beieinander liegen, woraus sich schließen lässt, dass Vorhersage und Simulation über die Zeit hinweg ähnlich polarisiert sind. Hier ist hervorzuheben, dass wie in Abschnitt 3.1 besprochen, keine eindeutige Polarisierung vorzufinden ist. Die Kurve liegt in allen Bereichen unter 0.65.

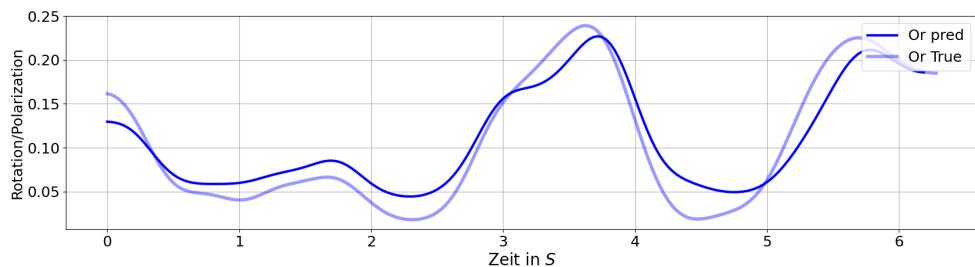


Abbildung 4.24: Rotationszustand der Simulation im Vergleich zur Vorhersage

Die obige Abbildung zeigt ebenfalls, dass die Zustände ähnlich verlaufen. Die Simulation und dessen Vorhersage rotieren demzufolge in ähnlicher Weise. Auch hier muss erwähnt werden, dass kein Rotationszustand erreicht wurde, da in allen Fällen $O_r < 0.65$ ist.



Abbildung 4.25: Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)

Das Betrachten der Laufwege der künstlichen Daten und der Simulation mittels approximierten Parametern zeigt, dass unterschiedliche Laufwege das Resultat dieses Experimentes sind. Es ist zu sehen, dass sich beide Simulationen ähneln. Es existiert kein Ausreißer, welche das eine Bild vom anderen abheben lässt.

4.2.2.2 Eigenes Modell

Für das eigene Modell gelten dieselben Ausgangssituationen wie für das metrische Modell. Für die Approximation wird RMD verwendet. Die Lernrate ist 0.02 und zum Anpassen der Lernrate wird ADAM verwendet. Hier werden pro Approximation nicht mehr als 2000 Iterationen durchlaufen.

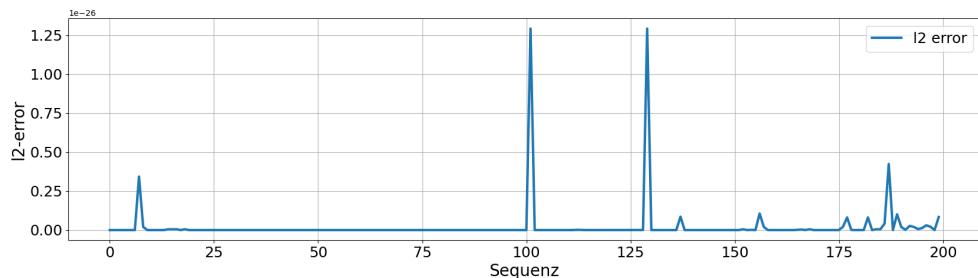


Abbildung 4.26: L2 Fehler für das eigene Modell

In der Abbildung 4.26 ist der Fehler des eigenen Modells zu sehen. Der Fehler liegt ausnahmslos bei null. Daraus kann geschlossen werden, dass die Vorhersage mit der künstlichen Simulation übereinstimmt.

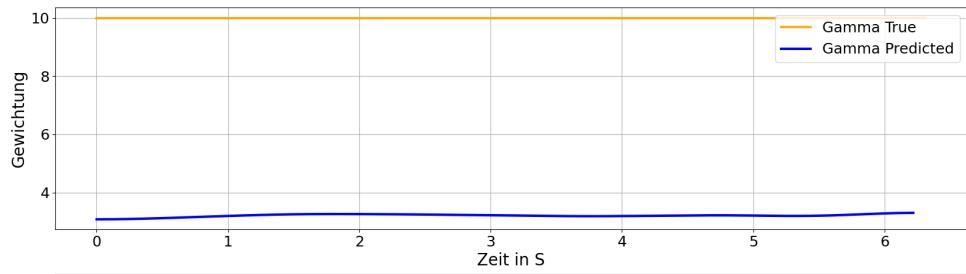


Abbildung 4.27: L2 Fehler für das eigene Modell

Ein Blick auf den Parameter Gamma zeigt, dass die Vorhersage ziemlich konstant bei ca. 3 liegt und somit nicht dem korrekten Parameter von 10 entspricht.

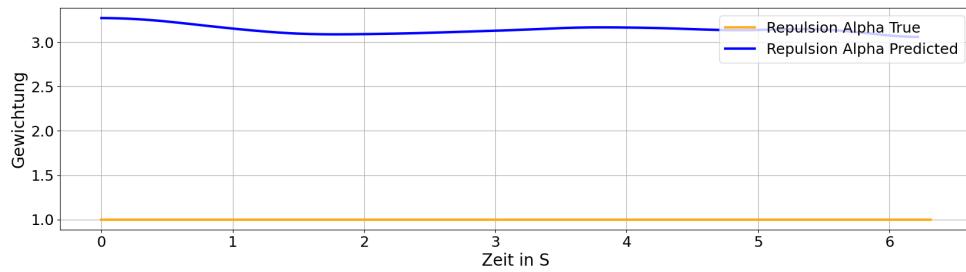


Abbildung 4.28: Parameter α_1 für das eigene Modell

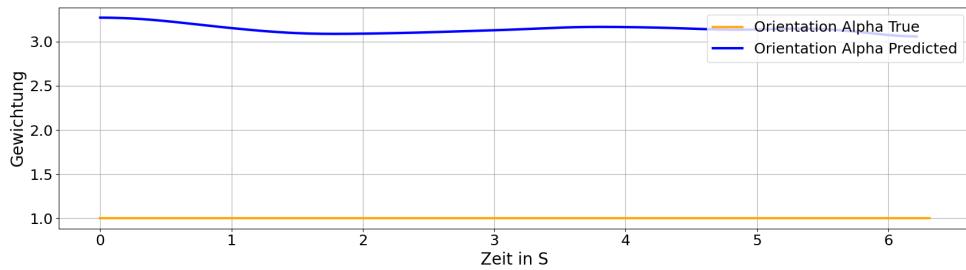


Abbildung 4.29: Parameter α_2 für das eigene Modell

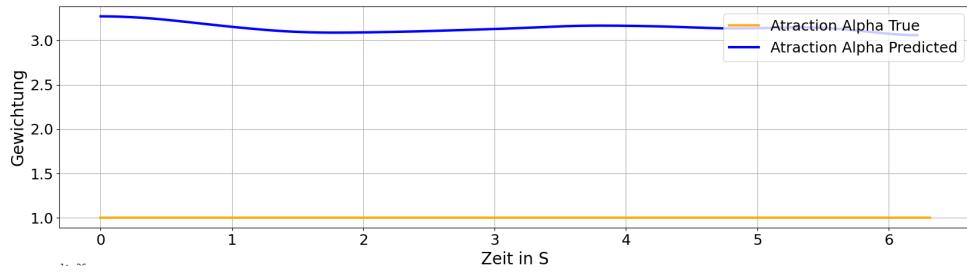


Abbildung 4.30: Parameter α_3 für das eigene Modell

Abbildungen 4.28 - 4.30 zeigen, dass auch hier die Parameter recht konstant vorhergesagt wurden, jedoch in allen drei Fällen mit einem zu hohen Wert. Aus Abschnitt 3.2.2 geht jedoch hervor, dass die Parameter nur Skalierungsfaktoren sind. Teilt man die Alphas durch 3 und multipliziert das Gamma mit 3, so erhält man in etwa die korrekten Parameter.

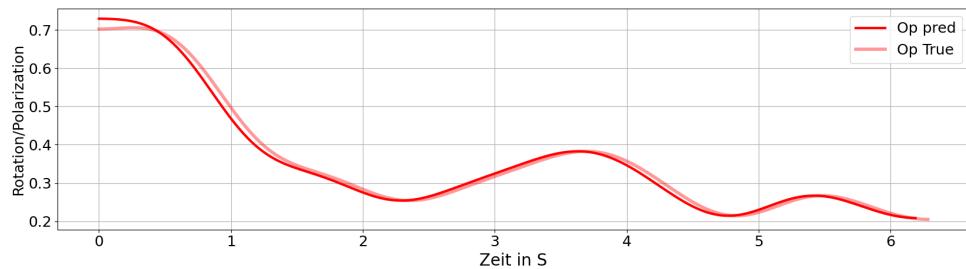


Abbildung 4.31: Polarisierungszustand für das eigene Modell

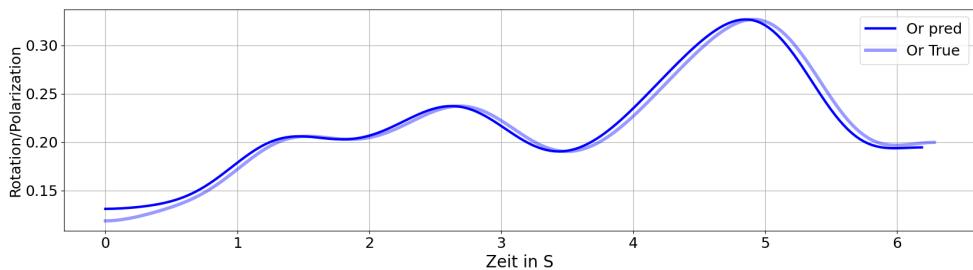


Abbildung 4.32: Rotationszustand für das eigene Modell

Ein Blick auf die Schwarmzustände zeigt auch hier, dass die Rotation und Polarisation für die künstlich erzeugten Daten und die der Approximation übereinstimmen.

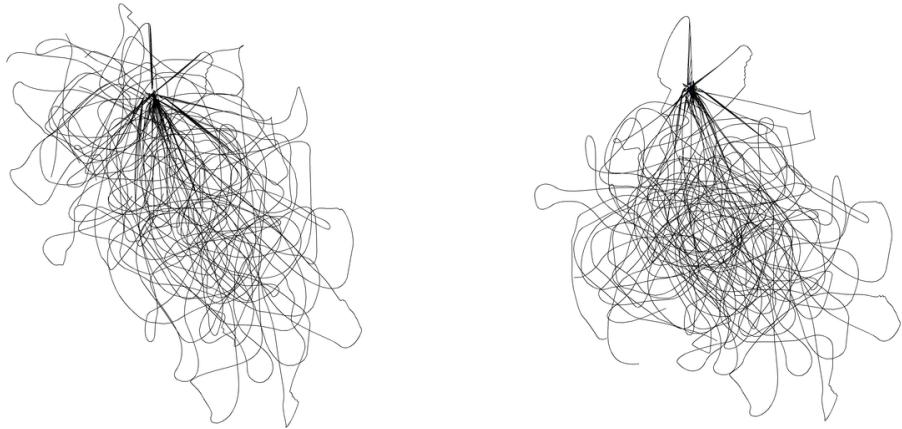


Abbildung 4.33: Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)

In Abbildung 4.50 sind die Trajektorien von diesem Teil des Experimentes zu sehen. Es sind durchaus Laufwege zu erkennen, die in beiden Graphen die gleichen sind. Allerdings existieren in beiden Graphen auch Laufwege, die in den jeweils anderen Graphen nicht vorhanden sind.

4.2.3 Diskussion

Die Parameterapproximation für konstante Parameter anhand von künstlichen Daten funktioniert für beide Modelle. Die Erkenntnis, die aus diesem Teil der Arbeit gezogen werden kann, ist, dass aus verschiedenen Parametersets dasselbe Verhalten erreicht werden kann. Anhand des erreichten Fehlers ist zu erkennen, dass RMD für die Approximation zuverlässig funktioniert. PSO minimiert den Fehler zwar, allerdings schwankt der Fehler für das metrische Boids Modell stark und weicht häufig von null ab. Im Mittel funktioniert die Approximation ausreichend gut für beide Modelle. Dies zeigen die Trajektorien, die sich in beiden Fällen sehr ähnlich sind sowie die Zustandsdiagramme, für die das Gleiche gilt.

4.2.4 Approximierung von variablen Parametern

In diesem Abschnitt wird die Approximation von Parametern anhand von künstlichen erzeugten Daten untersucht. Den Agenten werden wie in Abschnitt 4.2.2 zufällige Startpositionen und Richtungsvektoren zugewiesen. Die Simulation wird für einige Sequenzen durchlaufen, wobei zufällig gezogene Parameter das Verhalten der Agenten für die jeweils nächste Sequenz bestimmen.

4.2.4.1 Metrisches Boids Modell

Für dieses Modell wird wie schon zuvor PSO verwendet. Die Anzahl der Partikel pro Dimension ist hierbei 50. Nach maximal 300 Iterationen wird die Suche nach Parametern abgebrochen.

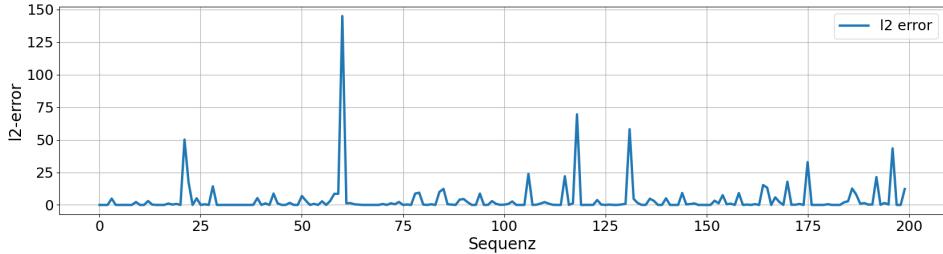


Abbildung 4.34: L2 Fehler für das Boids Modell mit variablen Parametern

In obiger Abbildung ist der Fehler pro Sequenz zu sehen. Im Vergleich zum Fehler für die Approximation von konstanten Parametern (siehe Abbildung 4.17) ist zu erkennen, dass hier der Fehler im Mittel geringer ausfällt. Die maximalen Spitzenwerte fallen jedoch höher aus. Da konstante Parameter eine geringere Herausforderung darstellen sollten, ist hier jedoch zu erwarten, dass der Fehler im Mittel höher ausfällt. Die Erklärung liefert hier die Arbeitsweise von PSO. Die zufällige Platzierung der Partikel im Suchraum ist ein treibender Faktor. Dazu kommt, dass nicht nur die Platzierung vom Zufall abhängt, sondern auch das Aktualisieren der Positionen der Partikel. Hieraus resultieren unterschiedliche Ergebnisse für die Parameterapproximationen. Insbesondere bedeutet das, dass beim mehrfachen Durchlauf desselben Experiments unterschiedliche Parameter und Fehler als Resultat auftreten.

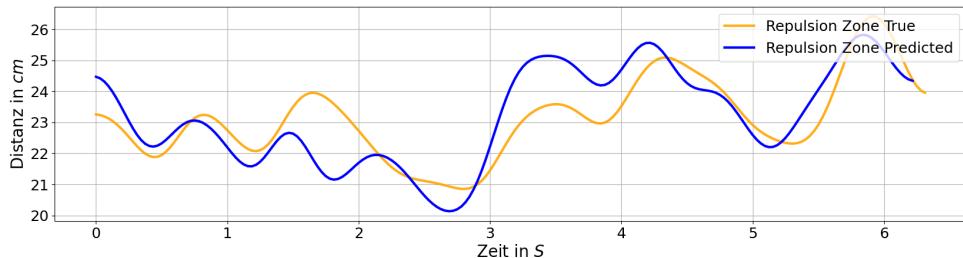


Abbildung 4.35: Abstandhalten Zone für das metrische Boids Modell mit variablen Parametern

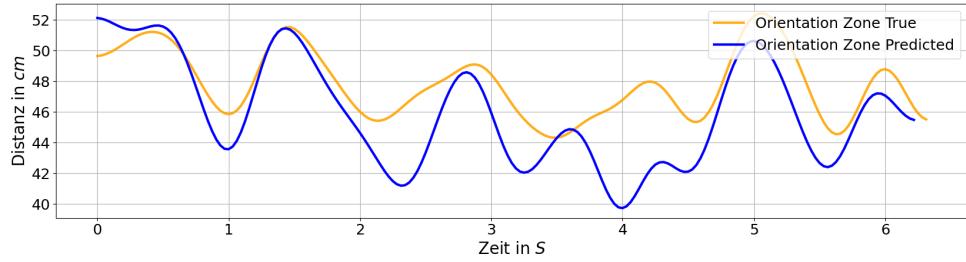


Abbildung 4.36: Orientierungszone für das metrische Boids Modell mit variablen Parametern

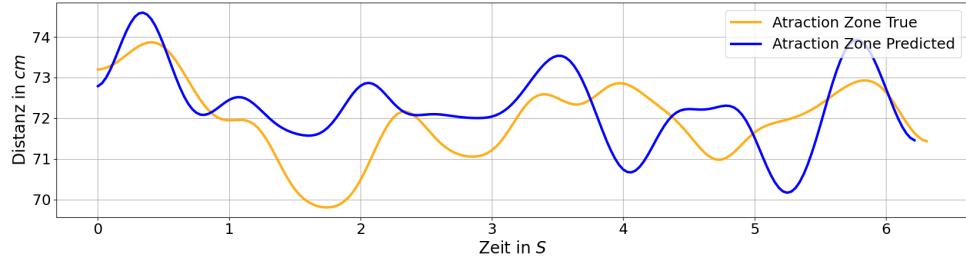


Abbildung 4.37: Attraktionszone für das metrische Boids Modell mit variablen Parametern

Die Vorhersage der Parameter für die drei Zonen ist in den Abbildungen 4.35 - 4.37 in Blau dargestellt, die der korrekten Werte in Gelb. Zu sehen ist, dass die Parameter sich über die Zeit hinweg verändern. Die Vorhersage ist hierbei relativ nahe bei den korrekten Werten. Die höchste Diskrepanz von ca. 8 cm befindet sich bei der Orientierungszone zum Zeitpunkt 4 Sekunden. Zudem kann man erkennen, dass die Zonen sich nicht überlappen. Das hat zur Folge, dass jede Zone relevant ist.

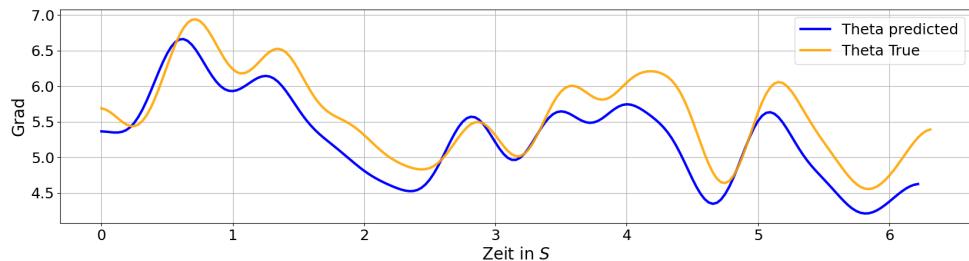


Abbildung 4.38: Einschlagwinkel für das metrische Boids Modell mit variablen Parametern

Der Einschlagwinkel ist in Abbildung 4.38 dargestellt. Hier ist zu sehen, dass

der Verlauf der Kurven sehr ähnlich ist. Die Verschiebung in Y-Richtung beträgt nicht mehr als 0.5 Grad.

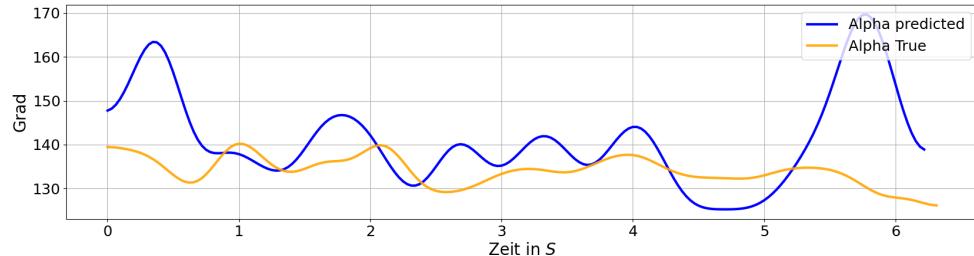


Abbildung 4.39: Blinde Zone für das metrische Boids Modell mit variablen Parametern

Der Winkel der blinden Zone der künstlichen Daten und dessen Vorhersage ist in Abbildung 4.39 zu sehen. Der größte Unterschied ist mit um die 30 Grad kurz vor Sekunde 6 zu sehen. Ansonsten verlaufen die Kurven eher unähnlich aber nahe beieinander. Die Relevanz der blinden Zone ist wie die anderen Zonen abhängig davon ob sich Agenten darin befinden.

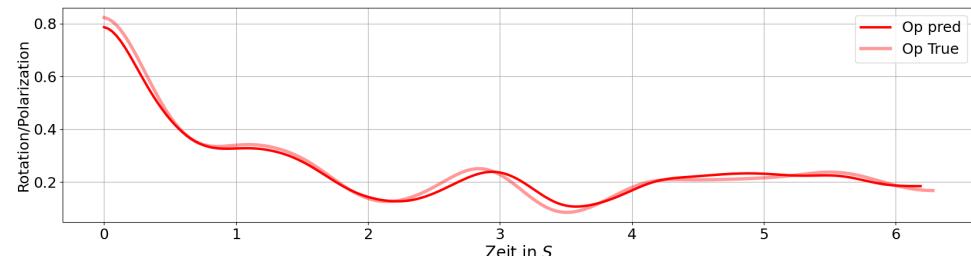


Abbildung 4.40: Zustand der Polarisierung für das Boids Modell mit variablen Parametern

Der Zustand der Polarisierung ist in obiger Abbildung zu sehen. Der Verlauf der Kurve ist fast identisch und daher kann hieraus geschlossen werden, dass das Verhalten der Agenten im Bezug auf die Polarisierung auch nahe beieinander liegt.

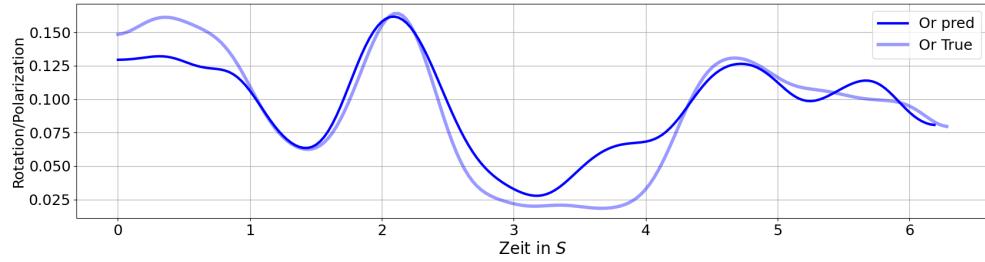


Abbildung 4.41: Rotationszustand für das Boids Modell mit variablen Parametern

Die Kurven für den Rotationszustand verlaufen relativ nahe beieinander. Anzumerken ist, dass der Wertebereich sehr niedrig ist und somit keine Rotation vorhanden ist.

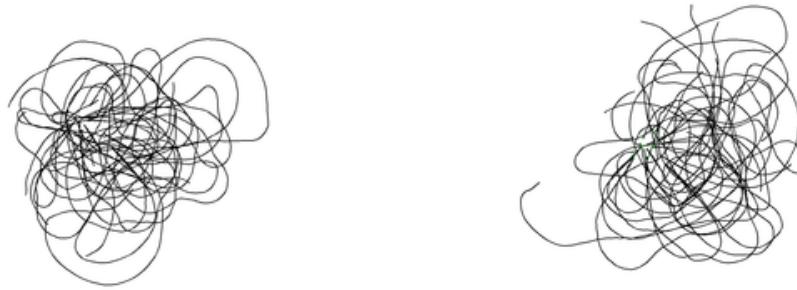


Abbildung 4.42: Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)

Auch hier ist wieder zu sehen, dass die Laufwege des Boids Modells nicht identisch sind. Die Trajektorien sind deutlich zu unterscheiden, ihr Verlauf ist dennoch ähnlich. Die Agenten bewegen sich in diesem Fall recht gleichmäßig im Raum. Es gibt keine Position, die von den Agenten ungewöhnlich oft besucht wird. Man kann wie auch schon zuvor erkennen, dass der Zusammenhang gegeben ist.

4.2.4.2 Eigenes Modell

Für das eigene Modell wird auch hier RMD genutzt mit der Lernrate 0.02 und ADAM zur Anpassung der Lernrate. Die Suche nach Parametern läuft für 2000 Iterationen.

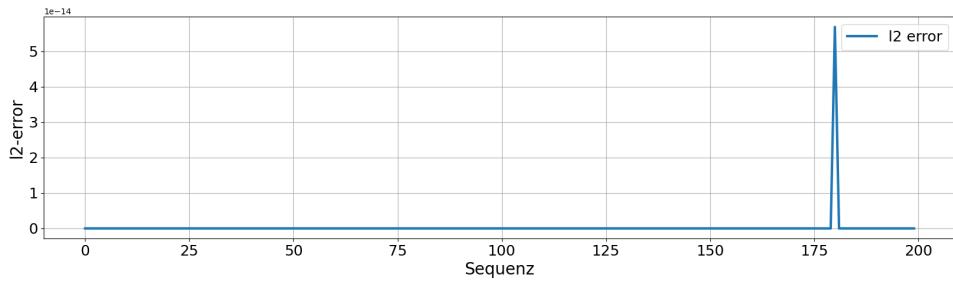


Abbildung 4.43: L2-Fehler für das eigene Modell mit variablen Parametern

Der Fehler für das eigene Modell ist hier ebenfalls niedriger als für das metrische Boids Modell. Der maximale Wert liegt hier bei $5e^{-14}$ was gleich null entspricht. Auch hier kann man sehen, dass die Approximation via RMD einen geringen Fehler produziert.

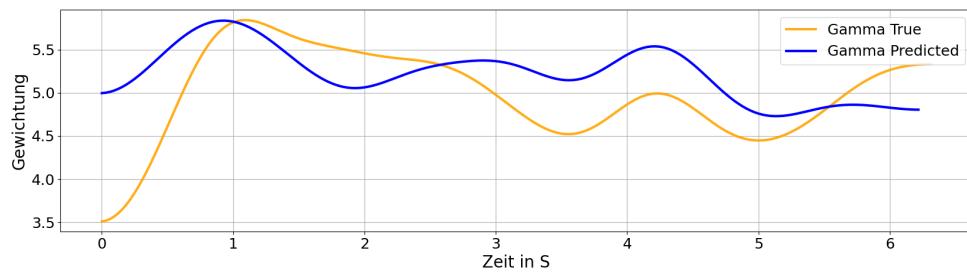


Abbildung 4.44: Parameter γ des eigenen Modells mit variablen Parametern

Die Approximation des Gammawertes zeigt, dass die Vorhersage relativ nahe bei dem korrekten Gammawert liegt.

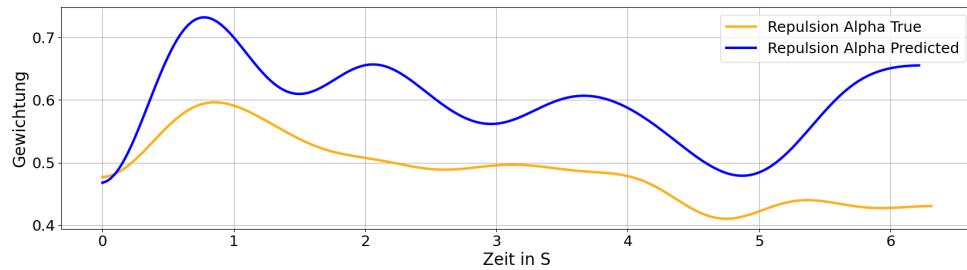


Abbildung 4.45: Parameter α_1 des eigenen Modells mit variablen Parametern

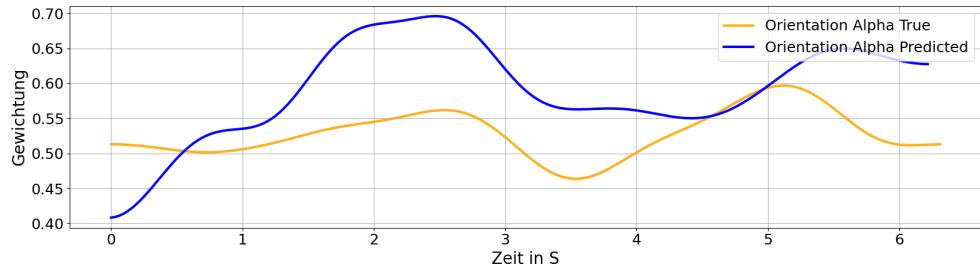


Abbildung 4.46: Parameter α_2 des eigenen Modells mit variablen Parametern

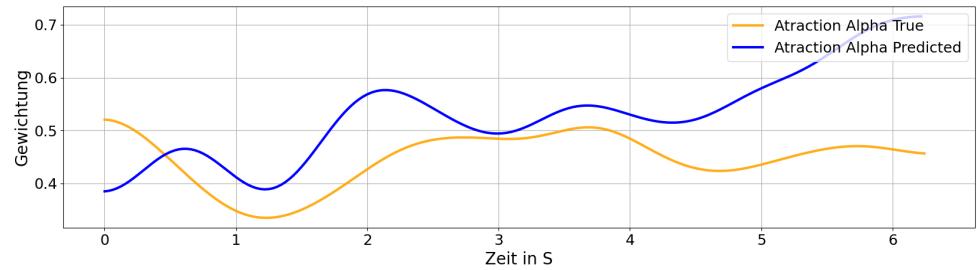


Abbildung 4.47: Parameter α_3 des eigenen Modells mit variablen Parametern

Die Alphaparameter, wie sie in den oberen drei Abbildungen zu sehen sind, zeigen einen jeweils unterschiedlichen Kurvenverlauf. Auch hier liegen diese nahe beieinander. Wie zuvor erwähnt, sind die Alphaparameter und der Gammaparameter Skalierungsfaktoren, wodurch unterschiedliche Parameterkonstellationen das gleiche Verhalten für die Simulation bedeuten können.

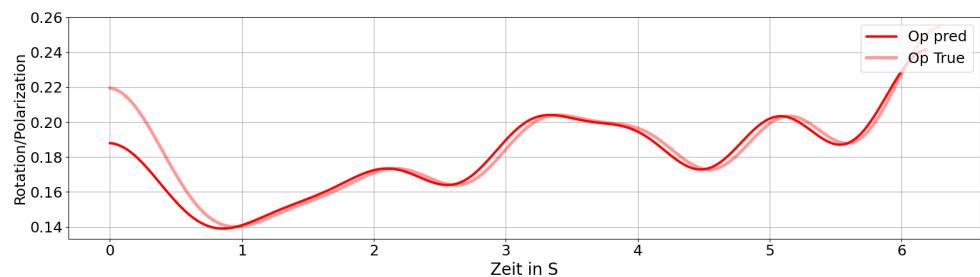


Abbildung 4.48: Polarisierungszustand für das eigene Modell mit variablen Parametern

Die Polarisierungszustände der künstlichen Daten und der Approximation liegen hier auch sehr nahe beieinander. Zu Beginn lässt sich ein Unterschied feststellen, der in etwa bei 0.03 liegt.

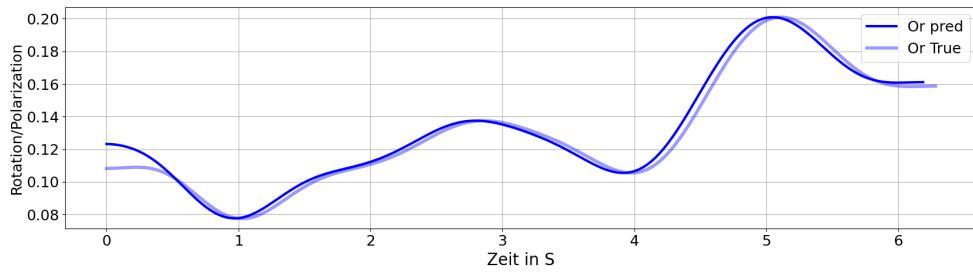


Abbildung 4.49: Rotationszustand für das eigene Modell mit variablen Parametern

Der Rotationszustand liegt auch für variable Parameter nahe beieinander. Auch hier kann nur zu Beginn ein Unterschied festgestellt werden, der bei ca 0.02 liegt.

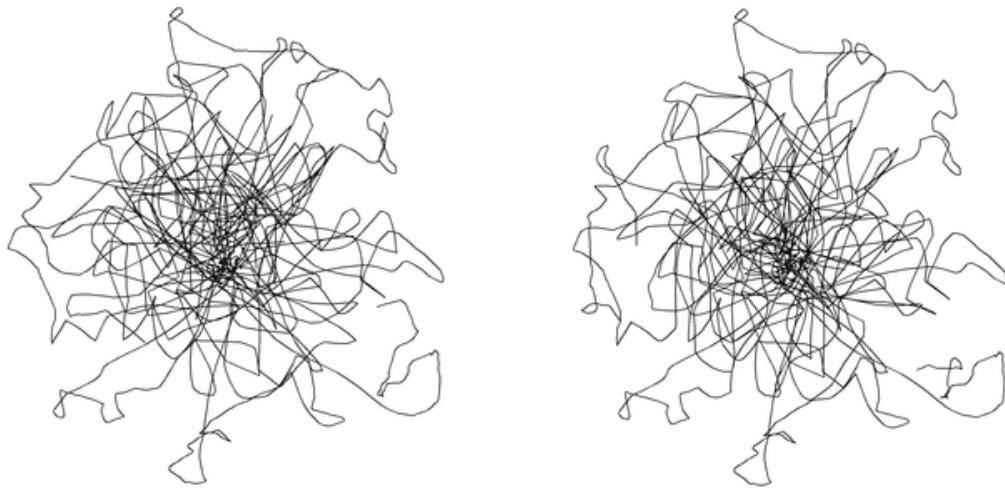


Abbildung 4.50: Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)

Die Pfade der Agenten sind hier ebenfalls sehr ähnlich. Es lässt sich erkennen, dass die Agenten einen recht starken Drang zum Mittelpunkt haben. Im Vergleich zum Modell von Boids sind die Pfade hier und auch schon im Experiment zuvor recht scharfkantig. Das metrische Boids Modell zeichnet sich durch einen geradlinigen Verlauf der Pfade aus. Der entscheidende Faktor ist hierbei der Einschlagswinkel, der in diesem Modell nicht vorhanden ist.

4.2.5 Diskussion

Die Approximation für die variablen Parameter zeigt für beide Modelle, dass die Zustände der Schwärme ähnlich verlaufen. Die exakten Parameter wurden

für beide Modelle nicht erreicht. Die Zustandsverläufe sind trotzdem sehr nahe beieinander. Es lässt sich zu diesem Zeitpunkt vermuten, dass eine Parameterbestimmung für die realen Trajektorien der Fische zu bewältigen ist. Zu erwarten ist, dass die Approximation anhand der Realdaten unterschiedliche Trajektorien zustande kommen. Das eigene Modell wird eher zentrierte Pfadwege besitzen mit durchaus kantigen Pfaden. Das metrische Boids Modell hingegen wird einen gleichmäßigen Verlauf aufweisen.

Ein Rotationszustand oder Polarisationszustand stellte sich mit den zufällig gezogenen Parametern nicht ein. Es gilt nun zu ergründen, ob die Zustände der realen Daten von den Modellen abgebildet werden können.

4.3 Approximation anhand von realen Daten

Die vorherigen Experimente haben gezeigt, dass aus der Approximation der Parameter für künstliche Daten Parameter resultieren, welche dieselben Zustände erzeugen können wir die der Simulation. In diesem Teil des Kapitels werden Parameter anhand der Trajektorien aus Abschnitt 4.1.3 der Fischschwärme approximiert. Dazu werden die beiden vorgestellten Modelle begutachtet.

Als Kostenfunktion wurde wie schon zuvor die l2-Kostenfunktion verwendet. Die nachfolgenden Bilder zeigen den Fehler über einen Zeitraum von 2000 Bildsequenzen, was in etwa 65 Sekunden entspricht. Die betrachtete Schwarmgröße beträgt 10 Fische wobei für beide Modelle dieselben Sequenzen approximiert werden.

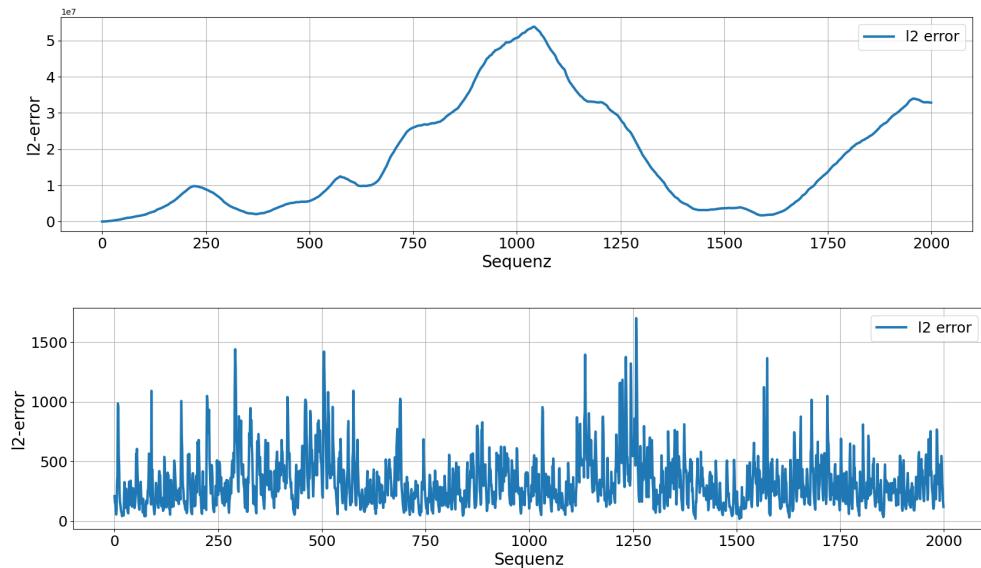


Abbildung 4.51: Kostenfunktion für das metrische Boids Modell (oben) und das eigene Modell (unten)

Beim betrachten des Fehlers erkennt man, dass der Fehler des metrischen Boids Modells die ersten 1000 Sequenzen stark ansteigt und bei ca. $5e^{-7}$ sein Maximum findet. Nach dem Maximum sinkt der Fehler wieder ab, bleibt jedoch in einem relativ hohen Wertebereich. Der Fehler des eigenen Modells hat sein Maximum bei 1250 und ist insgesamt in einem niedrigeren Wertebereich als der des metrischen Modells. Hieraus kann man schließen, dass das eigene Modell für die Parameterapproximation anhand von Realdaten besser geeignet ist als das metrische Boids Modell. Ein Blick auf die Zustände bestätigt diese Beobachtung.

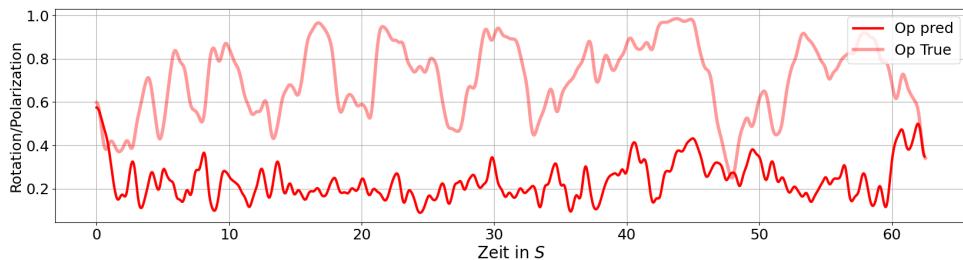


Abbildung 4.52: Kostenfunktion für das metrische Boids Modell (oben) und das eigene Modell (unten)

Der Zustand der Polarisierung des Schwarms und der Simulation mit approximierten Parametern unterscheiden sich sehr stark. Das Modell zeigt keine Polarisierung, da dessen Wertebereich nahezu durchgehend unter 0.65 liegt. Die Realdaten hingegen weisen in einigen Bereichen Polarisierung auf. Beispielsweise liegt der Wertebereich der Polarisierung für die Realdaten zwischen Sekunde 40 und 45 bei mehr als 0.65 und der Wertebereich der Rotation bei unter 0.35.

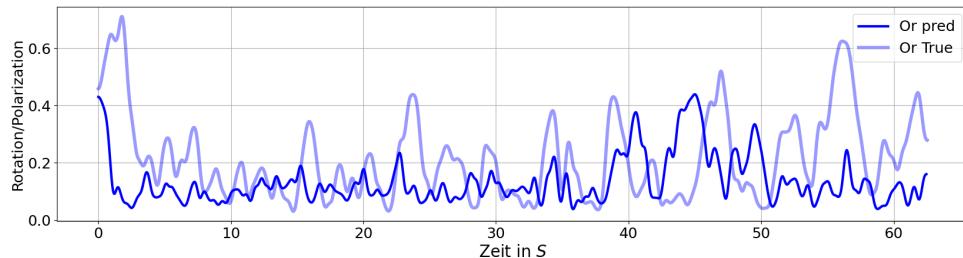


Abbildung 4.53: Rotationszustand der Realdaten mit 10 Fischen und des metrische Boids Modells

Auch der Rotationszustand, wie er in Abbildung 4.53 zu sehen ist zeigt, dass dieser von der Simulation nicht abgebildet wird. Zwar ist der Wertebereich näher beieinander, jedoch kann kein ähnlicher Verlauf ausgemacht werden. Hierbei findet Rotation der Realdaten ausschließlich in den ersten 1 – 4 Sekunden statt. Die nachfolgende Abbildung zeigt die Zustandsverläufe für das eigene Modell.

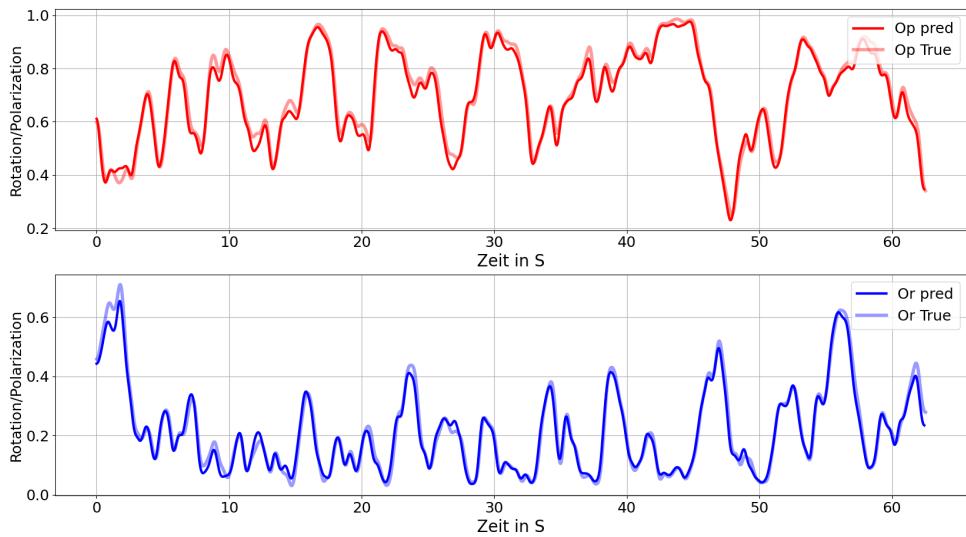


Abbildung 4.54: Polarisationszustand (oben) und Rotationszustand (unten) des eigenen Modells und des Schwarms der Größe 10 Fische

Zu erkennen ist, dass in beiden Fällen die Kurven sehr ähnlich verlaufen. Wie zuvor beschrieben findet Polarisation innerhalb der Realdaten statt. Somit kann das Modell Polarisation abbilden. Das Abbilden der Rotation ist hier nicht eindeutig zu sehen. Wie in Abschnitt 4.1.3 dargelegt, neigen die kleineren Schwärme weniger zur Rotation als die großen Schwärme. Demzufolge wird hierfür später ein größerer Schwarm betrachtet.

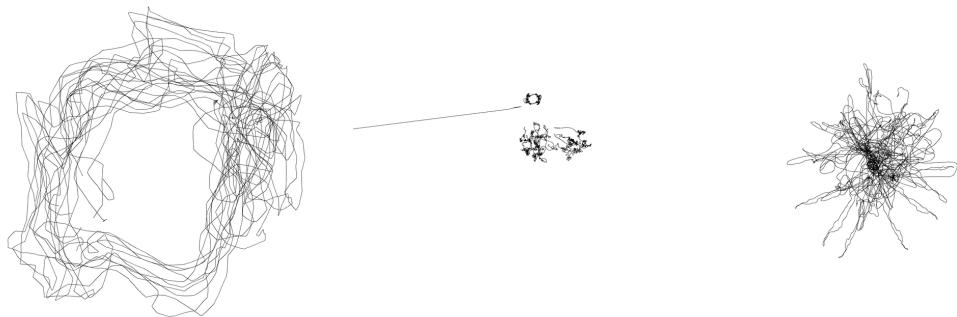


Abbildung 4.55: Trajektorien der Realdaten für 10 Fische (links), des metrischen Boids Modells (mitte) und des eigenen Modells (rechts)

Abbildung 4.55 zeigt die Trajektorien der Realdaten und der Modelle. Es ist festzustellen, dass die drei Trajektorien sehr unterschiedlich sind. Der echte Schwarm schwimmt kreisförmig um ein Zentrum herum.

Die Simulation des Boids Modells zeigt keine eindeutige Strukturierung. Ich frag mal Frau Bub zu sehen ist, dass sich ein Agent komplett vom Schwarm entfernt.

Die restlichen Agenten bleiben beieinander, entfernen sich jedoch nicht weit von ihrer Ausgangsposition.

Die Simulation des eigenen Modells zeigt eine Struktur, welche Zusammenhalt aufweist. Die Agenten haben einen starken Drang zur Mitte, werden allerdings immer wieder nach außen getrieben. Es findet ein Expandieren und Kollabieren der Struktur statt.

Nun stellt sich die Frage, wieso die Trajektorien der Realdaten keinen Rotationszustand erreichen, wenn diese doch eine eindeutige Rotation aufweist. Der Schwarm ist durch den Versuchsaufbau gezwungen, im Kreis zu schwimmen. Dadurch werden die Fische über eine längere Zeit eine Kreisbewegung ausführen. Das Rotationsdiagramm hingegen entsteht durch die Orientierung der Fische einer Momentaufnahme. Demnach kann die Orientierung der Fische in einem Moment eine schwache Ausprägung der Rotation besitzen, was über die Zeit hinweg dennoch in einer Rotation resultiert.

Das nachfolgende Experiment wird mit den Trajektorien von 60 Fischen durchgeführt, wobei diese eine stärkere Rotation aufweisen werden. Hierbei werden wie zuvor 2000 Sequenzen durchlaufen, was in etwa 65 Sekunden entspricht.

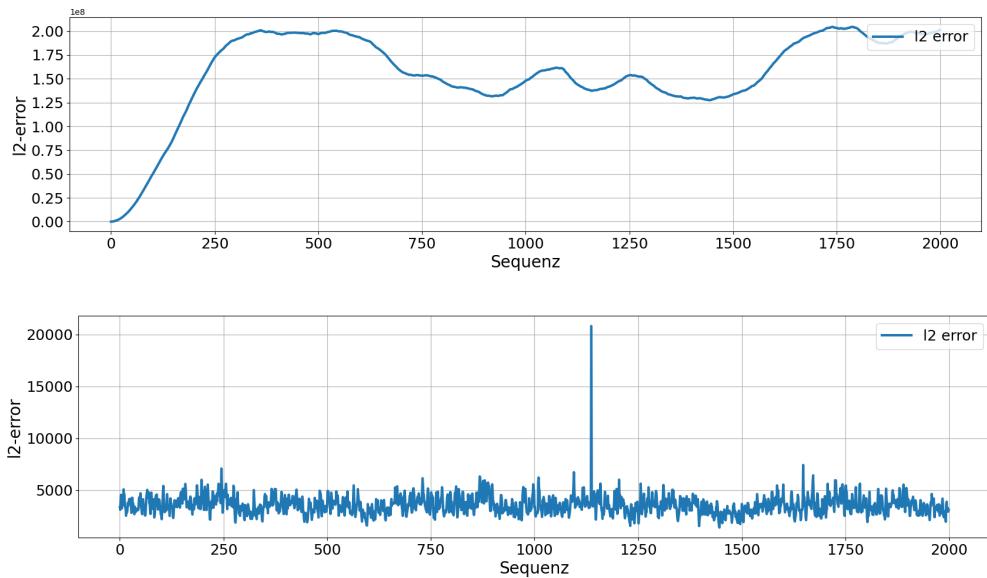


Abbildung 4.56: Kostenfunktion für das metrische Boids Modell (oben) und das eigene Modell (unten)

Auch hier ist wieder ein großer Unterschied bezüglich der Kosten zu sehen. Das Maximum der Kosten für das metrische Modell liegt bei $2e^8$ im Vergleich dazu liegt das Maximum des eigenen Models bei $2e^5$ und somit um den Faktor 1000 niedriger. Die Kosten des eigenen Modells sind im Durchschnitt fünfstellig, die des Modells von Boids weitestgehend achtstellig (bis auf die ersten Sequenzen). Die Zustandsdiagramme zeigen ein ähnliches Verhalten wie die Diagramme für den kleineren Schwarm.



Abbildung 4.57: Rotationszustand der Realdaten mit 60 Fischen und des metrische Boids Modells

Wie Abbildung 4.58 zeigt, passt sich die Kurve nicht der des Datensatzes an. Sie verläuft weitestgehend linear. Daraus kann man schließen, dass die Agenten der Simulation in unterschiedliche Richtungen schwimmen. Da so gut wie keine Polarisation stattfindet, kann auch keine Rotation stattfinden. Um zu rotieren, müssen die Agenten in gewisser Weise gleichgerichtet sein. Dies ist nur der Fall, wenn die Agenten gemeinsam in eine Richtung schwimmen. Dies zeigt das nächste Diagramm.



Abbildung 4.58: Rotationszustand der Realdaten mit 60 Fischen und des metrische Boids Modells

Auch hier findet bei der Simulation keinerlei Rotation statt (der Anfang sei ausgenommen). Die Trajektorien der Realdaten (transparent) zeigen hier jedoch eine deutliche Rotation. Vom Zeitpunkt 0 bis ca. 10 Sekunden ist $Or > 0.65$ und $Op < 0.35$ (Siehe Abbildung 4.57). Auch hier bildet das Modell das Verhalten der Realdaten nicht ab.

Das eigene Modell zeigt auch für eine größere Anzahl an Fischen ein ähnliches Verhalten wie für den kleineren Schwarm.



Abbildung 4.59: Polarisationszustand der Realdaten mit 60 Fischen und des eigenen Modells

Auch hier liegen die Kurven nahe beieinander. Wie beim kleineren Schwarm sind die Unterschiede hier marginal.

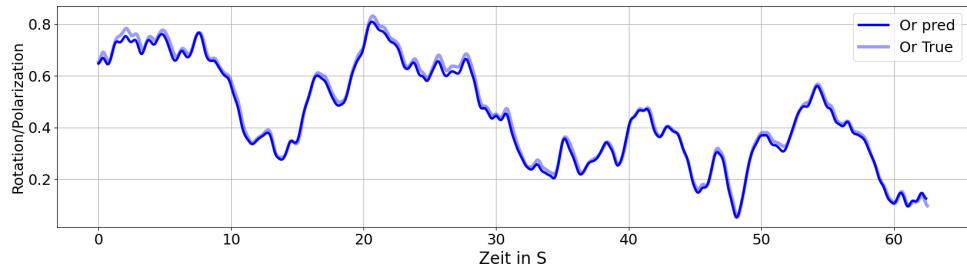


Abbildung 4.60: Rotationszustand der Realdaten mit 60 Fischen und des eigenen Modells

Gleiches gilt für den Rotationszustand, wie er oben zusehen ist. Auch hier lässt sich feststellen, dass das Modell die Zustände gut imitiert.

Ein Blick auf die Trajektorien zeigt die verschiedenen Verhaltensmuster der Daten und der Simulation.



Abbildung 4.61: Trajektorien der Realdaten für 60 Fische (links), des metrischen Boids Modells (mitte) und des eigenen Modells (rechts)

In der Abbildung 4.61 sind links die Trajektorien des Schwarms zusehen. Hier lässt sich feststellen, dass der Schwarm wie auch zuvor rotiert. Die Wege der Fische sind relativ ausgeglichen, wodurch sich die Bahnen gleichmäßig verteilen. Es existiert somit kein Bereich in dem sich die Fische ungewöhnlich oft aufhalten.

Das metrische Boids Modell ist mittig zu sehen und zeigt auch hier keinerlei Struktur. Die Agenten zeigen keinen Zusammenhalt, was durch die geradlinigen Trajektorien, welche aus dem Bild verschwinden, deutlich wird. Die Agenten sind somit in unterschiedliche Richtungen unterwegs.

Die Simulation für das eigene Modell (rechtes Bild) zeigt wieder die Eigenschaft des Expandieren und des Kollabierens. Das Zentrum wird von den Agenten häufiger besucht als die Außenbereiche.

4.3.1 Diskussion

Die Approximation der Parameter für die Modelle zeigt, dass das metrische Boids Modell schon bei den Diagrammen von den Realdaten zu unterscheiden ist. Es lassen sich keine Zustände imitieren und die Pfade der Simulation zeigen keinerlei Ähnlichkeit mit den Realdaten. Der direkte Vergleich der Fehlerfunktion zwischen dem metrischen Boids Modell und dem eigenen Modell zeigt, dass das eigene Modell einen geringeren Fehler pro Sequenz erreicht. Dies zeigt, dass das eigene Modell für die Approximation der Parameter für die Realdaten besser geeignet ist. Ob dies dem Approximierungsverfahren geschuldet ist oder sich das Modell nicht für Realdaten eignet, lässt sich hierbei nicht genau klären. Ein weiterer Kritikpunkt ist, dass kein Gesamtzusammenhalt der Agenten gewährleistet ist. Der Zusammenhalt wird ausschließlich über die Nachbarschaft erreicht. Dies ist ein Problem, wenn ein Agent keine weiteren Agenten in der entsprechenden Zone vorfindet. Daraus resultiert das Herauslösen des Individuums aus dem Schwarm, wie es in diesem Teil der Arbeit zu sehen war.

Das eigene Modell zeigt hinsichtlich der Zustandsdiagramme eine gute Anpassung an die echten Schwarmdaten. Es ist hierbei nicht von diesen zu unterscheiden. Die Trajektorien sprechen hierbei allerdings eine andere Sprache. Die Fische der echten Daten schwimmen in relativ ausgeglichner Weise im Becken und bevorzugen keinen Bereich. Die Agenten der Simulation des eigenen Modells bevorzugen das Massezentrum, welches durch die Gesamtheit der Agenten entsteht. Dies ist somit deren bevorzugter Aufenthaltsbereich und unterscheidet hiermit die Realdaten von den simulierten Daten. Teilt man einem Leihen mit, dass sich Fischschwärme in ausgeglichener Art durch das Wasser bewegen, so wird dieser anhand der Trajektorien sofort die Realdaten von den simulierten unterscheiden können.

Dass die Agenten des eigenen Modells zum Mittelpunkt getrieben werden, ist der Gewichtungsfunktion (siehe Abschnitt 3.2.2) geschuldet. Dies gewährleistet zum einen den Zusammenhalt der Individuen, zum anderen werden die Agenten wie ein Gummiband zurück zum Massezentrum getrieben, je stärker diese sich vom Zentrum entfernen. Es bräuchte eine Kraft die dem entgegenwirkt um eine stabile Rotation zu erreichen.

5 Fazit und Ausblick

Diese Masterthesis hatte das Ziel, Parameter für Modelle zu bestimmen, durch die Zustände eines echten Schwarms erreicht werden. Hierfür wurden zuerst zwei einfache Szenarien betrachtet, wodurch gezeigt wurde, dass Parameter für künstlich erzeugte Daten geschätzt werden können. Das erste Szenario befasste sich mit einem konstanten Parameterset, welches eine Sequenz an künstlichen Daten erzeugte. Dieses Experiment hat gezeigt, dass das Schätzen der Parameter die gleichen Zustände in Abhängigkeit der Zeit zur Folge hat. Die Parameter konnten für beide Modelle nicht exakt approximiert werden. Die hieraus resultierenden Trajektorien unterscheiden sich hingegen entweder marginal oder zeigen dieselbe Charakteristik auf.

Auch das zweite Experiment zeigte, dass künstliche Daten, welche durch Parameter, die sich über die Zeit hinweg ändern, approximieren lassen. Auch hier konnten die Zustände der künstlichen Daten durch die Approximation erreicht werden. In keinem der beiden Experimenten stellte sich ein Rotationszustand oder Polarisationszustand ein. Dies war allerdings auch nicht Ziel der Experimente.

Die Approximation der Parameter anhand Realdaten zeigte, dass die Modelle unterschiedlich geeignet sind. Das metrische Boids Modell kann die Zustände der Realdaten nicht imitieren. Das eigene Modell hingegen zeigt vergleichbare Zustände. Die Trajektorien beider Modelle sind nicht vergleichbar mit den Trajektorien der Realdaten. Während beim eigenen Modell die Agenten immer wieder in Richtung Mittelpunkt gezogen werden, zeigt das metrische Boids Modell keinerlei realistische Laufwege.

Zukünftige Arbeiten sollten sich in erster Linie auf die Parameterapproximation des Boids Modelles konzentrieren. So gut PSO für einige Probleme funktionieren mag, hat dieses Verfahren hier gezeigt, dass es die gewünschten Ergebnisse nicht produzieren kann. RMD ist in dieser Hinsicht ein zuverlässigeres Verfahren, welches der Approximation im Bezug auf Boids zugutekommen könnte. Das eigene Modell zeigt zwar Rotationseigenschaften im Diagramm, jedoch nicht innerhalb der Trajektorien. Das Modell von Boids besitzt zwar theoretisch die Möglichkeit zu rotieren, dies lies sich in den Experimenten jedoch nicht abbilden. Hier sollten zukünftige Arbeiten Anpassungen vornehmen. Für das Boidsmodell sollte der topologische Ansatz untersucht werden. Dadurch kann das Problem des Auseinanderdriftens, wie es in dem letzten Versuch der Fall war, vermieden werden. Das eigene Modell könnte eine weitere Gewichtungsfunktion erhalten, die sicherstellt, dass Agenten ab einer gewissen Distanz zum Mittelpunkt von diesem abgestoßen werden.

Literatur

- [1] M Ballerini u. a. „Interaction Ruling Animal Collective Behaviour Depends on Topological rather than Metric Distance: Evidence from a Field Study“. In: *Proceedings of the National Academy of Sciences of the United States of America* 105 (Feb. 2008), S. 1232–7. DOI: 10.1073/pnas.0711437105.
- [2] Iain D. Couzin u. a. „Collective memory and spatial sorting in animal groups.“ In: *Journal of theoretical biology* 218 1 (2002), S. 1–11.
- [3] Jimmy Ba Diederik P. Kingma. „Adam: A Method for Stochastic Optimization“. In: (Dez. 2014), S. 249–256.
- [4] G. Farnebäck. *Polynomial Expansion for Orientation and Motion Estimation*. Linköping studies in science and technology : Dissertation. Institut of Technology Linköping University. Department of Electrical Engineering, 2002. ISBN: 9789173734752. URL: https://books.google.de/books?id=1%5C_hmNQAACAAJ.
- [5] Gunnar Farnebäck. „Two-Frame Motion Estimation Based on Polynomial Expansion“. In: *Image Analysis*. Hrsg. von Josef Bigun und Tomas Gustavsson. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003, S. 363–370. ISBN: 978-3-540-45103-7.
- [6] James J. Gibson. *The Perception Of The Visual World*. Boston: Houghton Mifflin, 1950.
- [7] Florian Halboth und Flavio Roces. „The construction of ventilation turrets in Atta vollenweideri leaf-cutting ants: Carbon dioxide levels in the nest tunnels, but not airflow or air humidity, influence turret structure“. In: *PLoS ONE* 12.11 (2017). DOI: 10.1371/journal.pone.0188162.
- [8] Heiko Hamann. *Schwarmintelligenz*. Springer Spektrum Berlin,Heidelberg, 2019. ISBN: 978-3-662-58961-8.
- [9] Korbinian Kienle u. a. „Neutrophils self-limit swarming to contain bacterial growth in vivo“. In: *Science* 372.6548 (2021), eabe7729. DOI: 10.1126/science.abe7729. eprint: <https://www.science.org/doi/pdf/10.1126/science.abe7729>. URL: <https://www.science.org/doi/abs/10.1126/science.abe7729>.
- [10] Bonaventura Majolo und Pengzhen Huang. „Group living“. In: Dez. 2017. ISBN: 978-3-319-47829-6. DOI: 10.1007/978-3-319-47829-6_1865-1.
- [11] Charles C. Margossian. „A Review of automatic differentiation and its efficient implementation“. In: *CoRR* abs/1811.05031 (2018). arXiv: 1811.05031. URL: <http://arxiv.org/abs/1811.05031>.

- [12] Federico Marini und Beata Walczak. „Particle swarm optimization (PSO). A tutorial“. In: *Chemometrics and Intelligent Laboratory Systems* 149 (2015), S. 153–165. ISSN: 0169-7439. DOI: <https://doi.org/10.1016/j.chemolab.2015.08.020>. URL: <https://www.sciencedirect.com/science/article/pii/S0169743915002117>.
- [13] Kyle Fox Pankaj K. Agarwal Esther Ezra. *Geometric Optimization Revisited*. 2019. URL: https://doi.org/10.1007/978-3-319-91908-9_5.
- [14] Dhara Patel und Saurabh Upadhyay. „Optical Flow Measurement using Lucas Kanade Method“. In: *International Journal of Computer Applications* 61 (Jan. 2013), S. 6–10. DOI: 10.5120/9962-4611.
- [15] Craig Reynolds. „Steering Behaviors For Autonomous Characters“. In: (Juni 2002).
- [16] Craig W. Reynolds. „Flocks, Herds, and Schools: A Distributed Behavioral Model“. In: *SIGGRAPH Computer Graphics@bookGibson1950-GIBTPO-2*, author = James J. Gibson, publisher = Boston: Houghton Mifflin, title = *The Perception Of The Visual World*, year = 1950 21.4 (Juli 1987), S. 25–34. ISSN: 0097-8930. URL: <http://doi.acm.org/10.1145/37402.37406>.
- [17] Francisco Romero-Ferrero u. a. „idtracker.ai: Tracking all individuals in large collectives of unmarked animals“. In: *CoRR* abs/1803.04351 (2018). arXiv: 1803.04351. URL: <http://arxiv.org/abs/1803.04351>.
- [18] Yuhui Shi und B.Gireesha Obaiahnahhatti. „A Modified Particle Swarm Optimizer“. In: Bd. 6. Juni 1998, S. 69–73. ISBN: 0-7803-4869-9. DOI: 10.1109/ICEC.1998.699146.
- [19] Nishitha Shrinath, Bidarakere Rangaswamy und Bathula Sreenivas Reddy. „Biomimics“. In: *IJMTER* 3 (Apr. 2016), S. 463–467.
- [20] Kolbjørn Tunstrøm u. a. „Collective States, Multistability and Transitional Behavior in Schooling Fish“. In: *PLoS Computational Biology* 9 (2013).
- [21] Dongshu Wang, Dapei Tan und Lei Liu. „Particle swarm optimization algorithm: an overview“. In: *Soft Computing* 22 (Jan. 2018). DOI: 10.1007/s00500-016-2474-6.

Abbildungsverzeichnis

1	Polarisationszustand (links) und Rotationszustand (rechts) der Realdaten mit 60 Fischen und des eigenen Modells	IV
2	Polarisationszustand (links) und Rotationszustand (rechts) der Realdaten mit 60 Fischen und des metrische Boids Modells	IV
3	Trajektorien der Realdaten für 60 Fische (links), des metrischen Boids Modells (mitte) und des eigenen Modells (rechts)	IV
2.1	Verhaltensmuster nach Reynolds, entnommen aus [15]	3
2.2	Metrisch und Topologische Distanzen, entnommen aus [8]	4
2.3	Gradientenabstieg visualisiert entnommen von http://neuralnetworksanddeeplearning.com/chap1.html besucht am 24.11.2021	5
2.4	Kettenregel als Graph	7
2.5	Flowchart des PSO entnommen aus [21]	10
2.6	Bewegung des Pixels p von Zeitpunkt t zu Zeitpunkt $t + \Delta t$ entnommen aus [14]	11
3.1	Zustände von Fischschwämen , entnommen aus [20]	14
3.2	Zonen im metrischen Modell eines Agenten nach [2], Abbildung entnommen aus selbigen Paper	16
3.3	Skalierungsfunktionen für die verschiedenen Richtungsvektoren .	19
4.1	Versuchsaufbau für die Aufnahme der Fischschärme entnommen aus https://idtrackerai.readthedocs.io/en/latest/setups.html (besucht am 24.11.2021)	22
4.2	Tracking anhand einer Simulation, horizontaler Verlauf links, rotierender Verlauf rechts	23
4.3	Ergebnisse der Trackingmethode, polarisierte Bewegung	24
4.4	Ergebnisse der Trackingmethode, rotierende Bewegung	24
4.5	Ablaufdiagramm des Trackingprozesses, entnommen aus Romero-Ferrero u. a. [17]	25
4.6	Netzwerkarchitekturen zur Klassifikation von sich berührenden und separaten Individuen entnommen aus Romero-Ferrero u. a. [17]	26
4.7	Netzwerkarchitekturen zur) entnommen aus Romero-Ferrero u. a. [17]	26

4.8	Auswertung der Trackinggenauigkeit entnommen aus Romero-Ferrero u. a. [17]	27
4.9	Mittlere Geschwindigkeit des Fischschwärms der Größe 10	28
4.10	Mittlere Geschwindigkeit des Fischschwärms der Größe 60	28
4.11	Mittlere Geschwindigkeit des Fischschwärms der Größe 100	29
4.12	Zustände des Fischschwärms der Größe 10	29
4.13	Zustände des Fischschwärms der Größe 60	29
4.14	Zustände des Fischschwärms der Größe 100	29
4.15	PSO vs. RMD anhand von Boids	31
4.16	PSO vs. RMD Anhand des eigenen Modells	32
4.17	L2-Fehler der konstanten Parametervorhersage	33
4.18	Vorhergesagte Abstandhalten-Zonen in Blau vs korrekte Zonen in Orange	33
4.19	Vorhergesagte Orientierungs-Zonen in Blau vs korrekte Zonen in Orange	34
4.20	Vorhergesagte Zusammenhalten-Zonen in Blau vs korrekte Zonen in Orange	34
4.21	Vorhergesagter Einschlagswinkel in Blau vs korrekter Winkel in Orange	35
4.22	Vorhergesagte Winkel der blinden Zone in Blau vs korrekter Winkel in Orange	35
4.23	Polarisierungszustand der Simulation im Vergleich zur Vorhersage .	36
4.24	Rotationszustand der Simulation im Vergleich zur Vorhersage .	36
4.25	Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)	37
4.26	L2 Fehler für das eigene Modell	37
4.27	L2 Fehler für das eigene Modell	38
4.28	Parameter α_1 für das eigene Modell	38
4.29	Parameter α_2 für das eigene Modell	38
4.30	Parameter α_3 für das eigene Modell	39
4.31	Polarisierungszustand für das eigene Modell	39
4.32	Rotationszustand für das eigene Modell	39
4.33	Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)	40
4.34	L2 Fehler für das Boids Modell mit variablen Parametern	41

4.35 Abstandhalten Zone für das metrische Boids Modell mit variablen Parametern	41
4.36 Orientierungszone für das metrische Boids Modell mit variablen Parametern	42
4.37 Attraktionszone für das metrische Boids Modell mit variablen Parametern	42
4.38 Einschlagwinkel für das metrische Boids Modell mit variablen Parametern	42
4.39 Blinde Zone für das metrische Boids Modell mit variablen Parametern	43
4.40 Zustand der Polarisierung für das Boids Modell mit variablen Parametern	43
4.41 Rotationszustand für das Boids Modell mit variablen Parametern	44
4.42 Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)	44
4.43 L2-Fehler für das eigene Modell mit variablen Parametern	45
4.44 Parameter γ des eigenen Modells mit variablen Parametern	45
4.45 Parameter α_1 des eigenen Modells mit variablen Parametern	45
4.46 Parameter α_2 des eigenen Modells mit variablen Parametern	46
4.47 Parameter α_3 des eigenen Modells mit variablen Parametern	46
4.48 Polarisierungszustand für das eigene Modell mit variablen Parametern	46
4.49 Rotationszustand für das eigene Modell mit variablen Parametern	47
4.50 Trajektorien der künstlich erzeugten Daten (links) und der mittels approximierten Parametern (rechts)	47
4.51 Kostenfunktion für das metrische Boids Modell (oben) und das eigene Modell (unten)	48
4.52 Kostenfunktion für das metrische Boids Modell (oben) und das eigene Modell (unten)	49
4.53 Rotationszustand der Realdaten mit 10 Fischen und des metrische Boids Modells	49
4.54 Polarisationszustand (oben) und Rotationszustand (unten) des eigenen Modells und des Schwarms der größte 10 Fische	50
4.55 Trajektorien der Realdaten für 10 Fische (links), des metrischen Boids Modells (mitte) und des eigenen Modells (rechts)	50
4.56 Kostenfunktion für das metrische Boids Modell (oben) und das eigene Modell (unten)	51

4.57	Rotationszustand der Realdaten mit 60 Fischen und des metrische Boids Modells	52
4.58	Rotationszustand der Realdaten mit 60 Fischen und des metrische Boids Modells	52
4.59	Polarisationszustand der Realdaten mit 60 Fischen und des eige- nen Modells	53
4.60	Rotationszustand der Realdaten mit 60 Fischen und des eigenen Modells	53
4.61	Trajektorien der Realdaten für 60 Fische (links), des metrischen Boids Modells (mitte) und des eigenen Modells (rechts)	53
A.1	Zustände Boids für konstante Parameter	XIII
A.2	Zustände Boids für variable Parameter	XIII
A.3	Zustände eigenes Modell für konstante Parameter	XIV
A.4	Zustände eigenes Modell für variable Parameter	XIV
A.5	Zustände Boids der Schwarmgröße 10	XV
A.6	Zustände Boids der Schwarmgröße 60	XV
A.7	Zustände eigenes Modell der Schwarmgröße 10	XVI
A.8	Zustände eigenes Modell der Schwarmgröße 60	XVI

Tabellenverzeichnis

4.1 Vergleich der Laufzeiten zwischen PSO und RMD	32
---	----

Anhänge

A Weitere Abbildungen

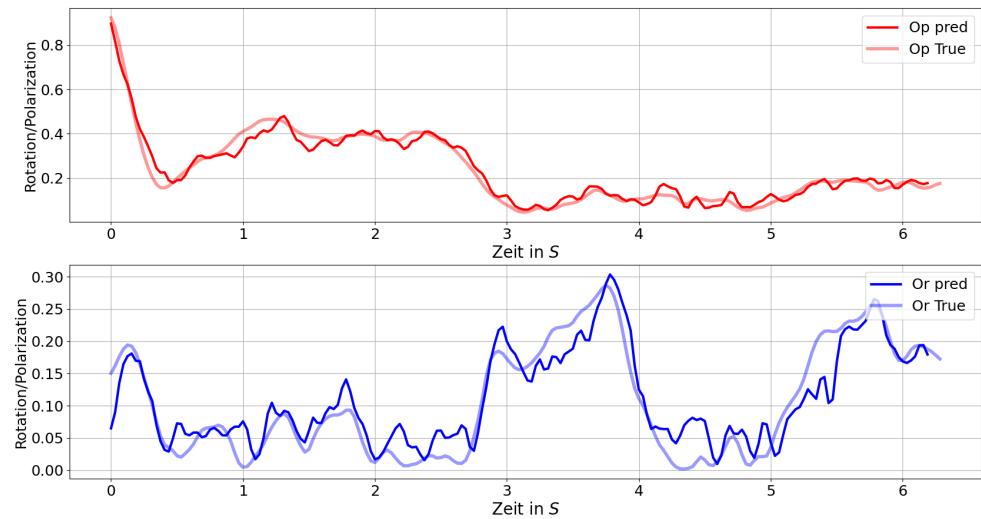


Abbildung A.1: Zustände Boids für konstante Parameter

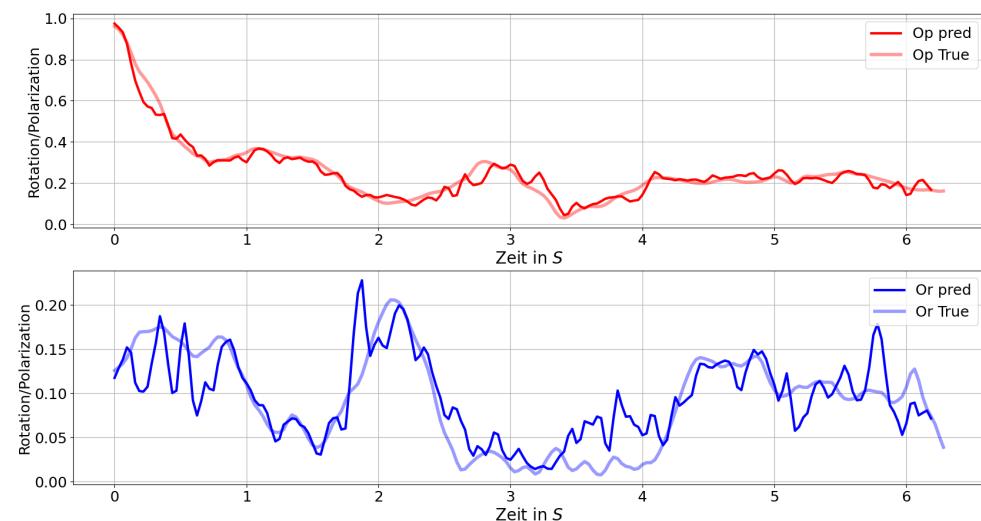


Abbildung A.2: Zustände Boids für variable Parameter

ANHANG A. WEITERE ABBILDUNGEN

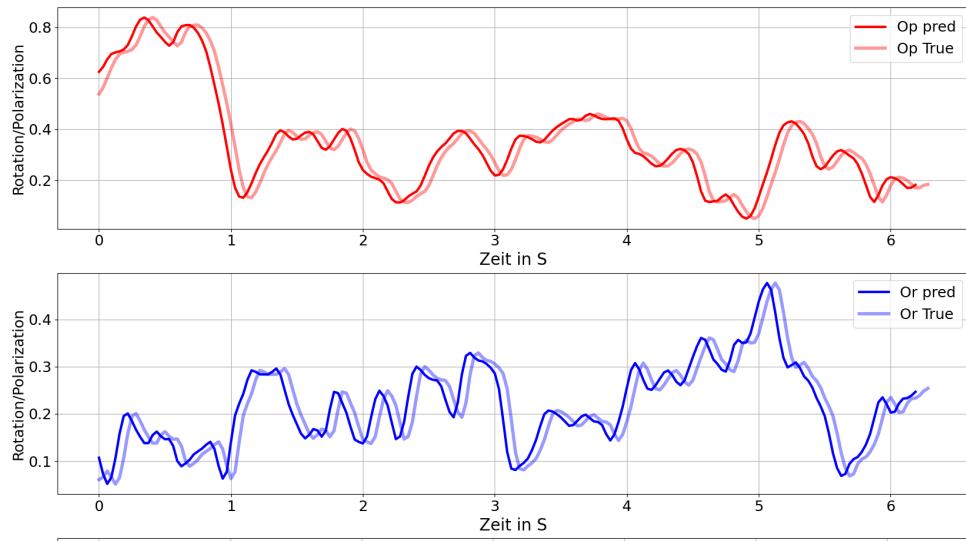


Abbildung A.3: Zustände eigenes Modell für konstante Parameter

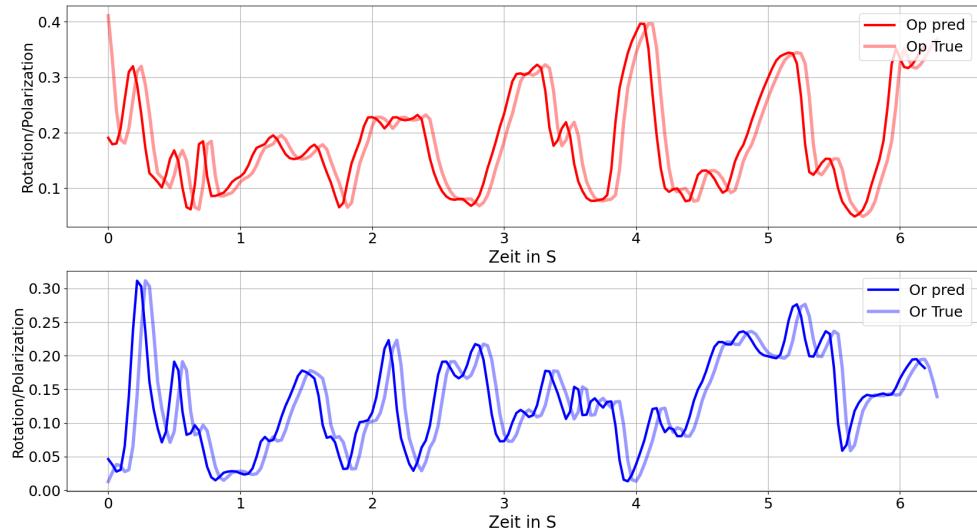


Abbildung A.4: Zustände eigenes Modell für variable Parameter

ANHANG A. WEITERE ABBILDUNGEN

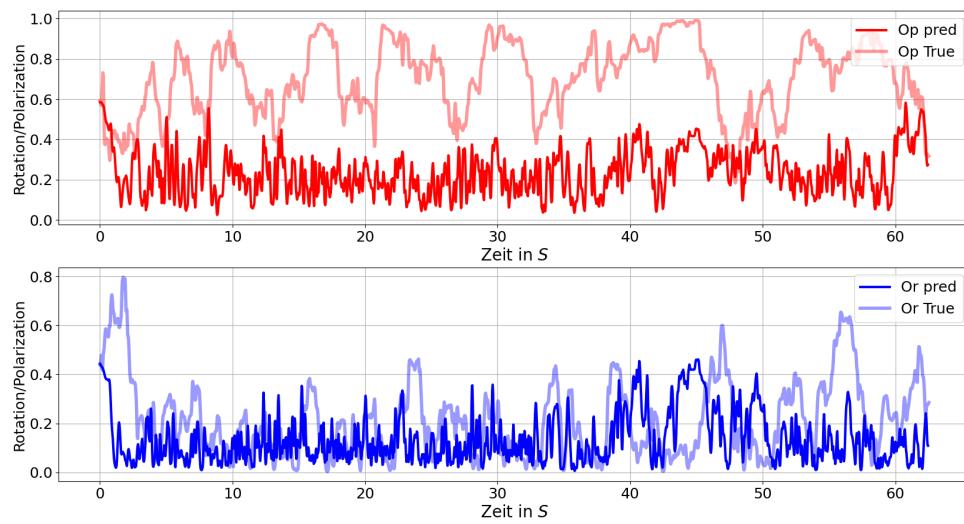


Abbildung A.5: Zustände Boids der Schwarmgröße 10

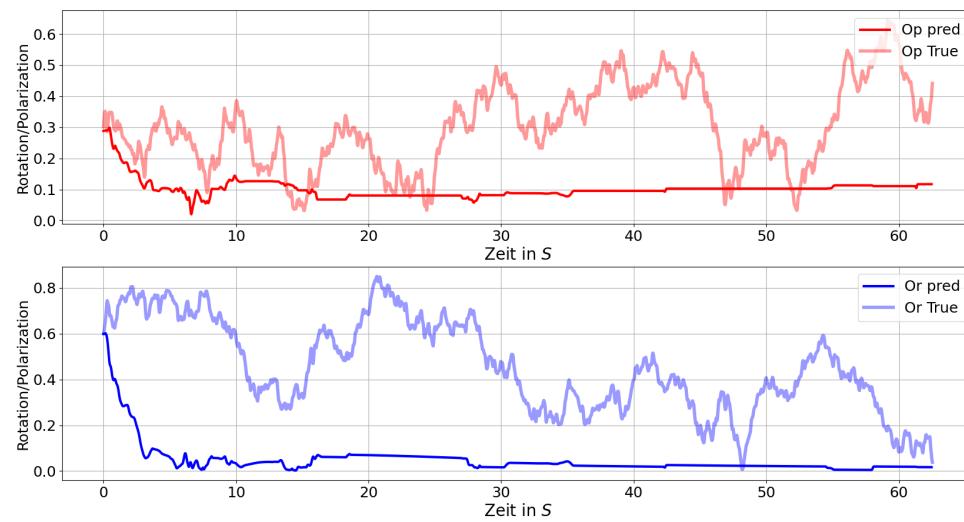


Abbildung A.6: Zustände Boids der Schwarmgröße 60

ANHANG A. WEITERE ABBILDUNGEN

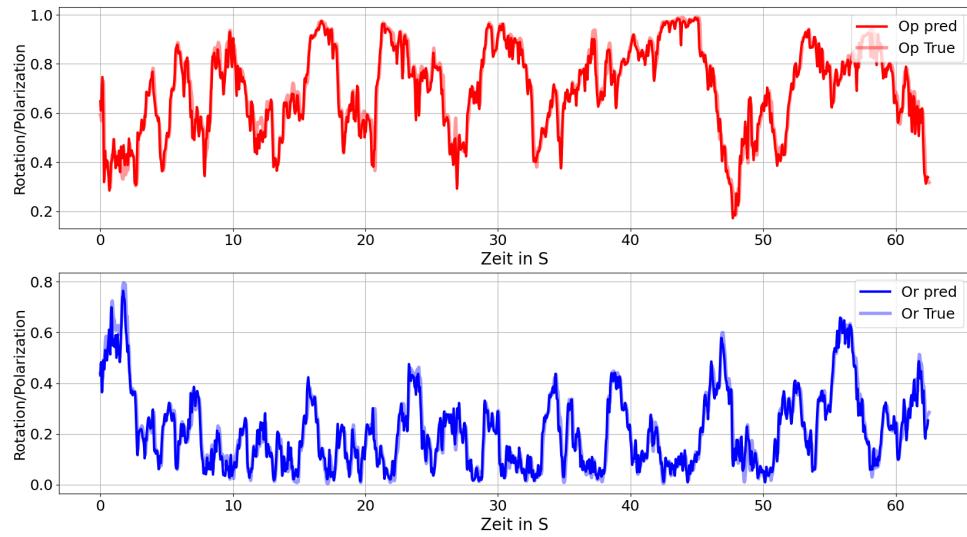


Abbildung A.7: Zustände eigenes Modell der Schwarmgröße 10

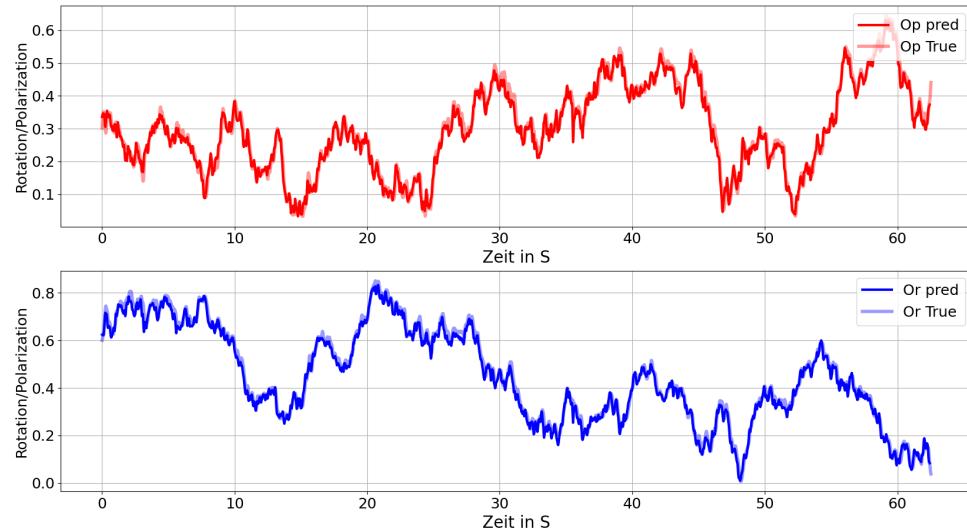


Abbildung A.8: Zustände eigenes Modell der Schwarmgröße 60