Hochschule Konstanz
Technik, Wirtschaft und Gestaltung

INSTITUT FÜR OPTISCHE SYSTEME

Modellierung von Schwarmverhalten Untersuchung von Methoden zur Parameter Approximation

Simon Christofzik

Konstanz, 14. August 2020

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: 17. Februar 2020
Abgabedatum: 14. August 2020

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 selbständig 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, 14. August 2020

(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:	14. August 2020
Schlagworte:	Boids, Collective behaviour, Reverse mode differentiation, Particle swarm optimization

BLABLABLA

Danksagung

blablabla

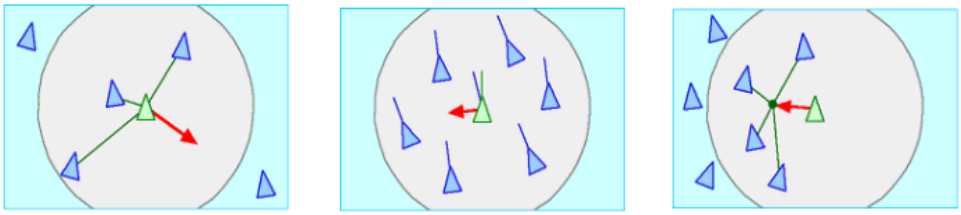
Inhaltsverzeichnis

1 Grundlagen

1.1 Modelle

1.1.1 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.



(a) (b) (c)

Abbildung 1.1: Verhaltensregeln nach Reynolds entnommen aus (C. Reynolds, 2002)

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 1.1 (a) ist zu sehen wie sich der Agent in Grün von den anderen Agenten im Radius 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 einem bestimmten Radius O. Hierzu wird über die Richtungen welche die Nachbarn innerhalb des Radius O schwimmen gemittelt.

c) **Zusammenbleiben**
Der Agent hat das Ziel beim Schwarm zu bleiben. Dazu schwimmt er in das Zentrum der Fische im Radius A.

Zu beachten ist, dass Reynolds in seinem Paper nur die Idee für ein Modell gibt. Ein expliziter Algorithmus wird jedoch nicht definiert C. W. Reynolds (1987).

Das Modell nach Reynolds ist rein Metrisch, jedoch hat eine Studie von Ballerini Ballerini et al. (2008) gezeigt, dass insbesondere Stare nur die sechs bis sieben nächsten Nachbarn, unabhängig von deren Distanz wahrnehmen. Dies wird als topologischer Ansatz bezeichnet.

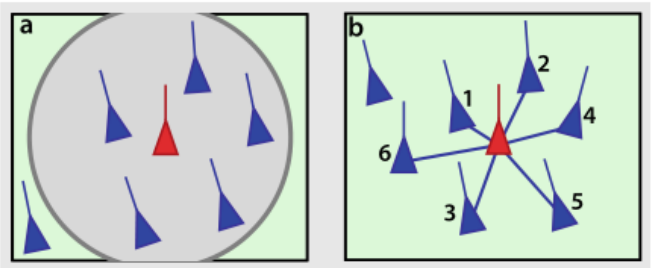


Abbildung 1.2: Metrisch und Topoligische Distanzen, entnommen aus (Hamann, 2019)

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. Beispielsweiße ist ein Angriff auf ein Kollektiv so ein Szenario, bei dem einzelne Fische 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, welches robuster durch das topoligische Modell als durch das metrische Modell erreicht wird.

1.2 Kombinatorische Optimierung

Optimum (unter den gegebenen Voraussetzungen, im Hinblick auf ein Ziel) höchstes erreichbares Maß, höchster erreichbarer Wert. DUDEN

Kombinatorische Optimierung bezeichnet das Auffinden einer oder meherer Teilmengen aus einer Grundmenge an Variablen welche bezüglich einer Kostenfunktion 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 (2019)

Mathematisch kann man ein Optimierungsproblem bezüglich einer Kostenfunktion folgendermaßen definieren:

Sei $f : \mathbb{R}^D \to \mathbb{R}$ 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.

1.3 Gradientenabstieg

Der Gradientenabstieg ist eine Methode um das Minimum einer Funktion zu berechnen.

1.3.1 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. Hierbei wird über die Steigung an einer Stelle ermittelt, in welche Richtung das Minimum oder Maximum liegt. Zu erwähnen ist, dass beim Gradientenabstieg nicht sichergestellt werden kann, dass das Minimum auch ein globales und nicht nur ein lokales Minimum ist (gleiches gillt für das Maximum). Ausgenommen hiervon sind konvexe Optimierungsprobleme. Bei dieser Art ist jedes lokale Minimum/Maximum auch ein Globales.

Für eine differenzierbare Funktion, welche in numerischer Form vorliegt ist Reverse Mode Differentiation (RMD) ein Algorithmus um deren Ableitung zu berechnen. Grundbaustein hierfür ist die Kettenregel.

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. Dann sieht die Kettenregel hierfür wie folgt aus:

$$\frac{\delta z}{\delta t} = \frac{\delta z}{\delta f_1}\frac{\delta f_1}{\delta t} + \frac{\delta z}{\delta f_2}\frac{\delta f_2}{\delta t} \tag{1.1}$$

Dies kann graphisch folgendermaßen dargestellt werden:

Dies graphische Darstellung einer Funktion wie sie in Gleichung 1.3 dargestellt ist, kann für jede differenzierbare Funktion angewandt werden.

Betrachten wir folgende Funktion:

$$f(x_1, x_2) = x_1 + x_2 sin(x_1) + ln(x_2) \tag{1.2}$$

Dies lässt sich komponentenweiße schreiben, wodurch sich folgender Graph ergibt:

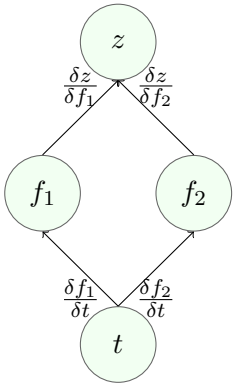
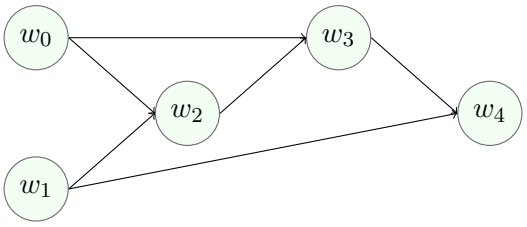


Abbildung 1.3: Kettenregel als Graph

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



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

$$
\begin{aligned}
&w_0 = 2 \\
&w_1 = 3 \\
&w_2 = w_1 sin(w_0) && \frac{\delta w_2}{\delta w_1} = sin(w_0) = 0.909 && \frac{\delta w_2}{\delta w_0} = 3cos(w_0) && = -1.24 \\
&w_3 = w_1 + w_2 && \frac{\delta w_3}{\delta w_1} = 1 && \frac{\delta w_3}{\delta w_2} = 1 \\
&w_4 = z = w_3 + ln(w_1) && \frac{\delta w_4}{\delta w_3} = 1 && \frac{\delta w_4}{\delta w_1} = \frac{1}{w_1} && = \frac{1}{3}
\end{aligned}
$$

Durch die Kettenregel kann nun $\frac{\delta w_4}{\delta w_0}$ und $\frac{\delta w_4}{\delta w_1}$ berechnet werden. Durch den Graphen und dem *Forwardpass* sind die Abhängigkeiten und deren Ableitungen sowie Werte bekannt. 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.

1.4 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 hierbei ist, dass das Optimum einer nicht ableitbaren Funktion bestimmt werden kann.

Wie der Name PSO vermuten lässt, handelt es sich hierbei um einen Algorithmus, der auf Schwarmverhalten setzt. Jeder einzelne Agent wird als Partikel bezeichnet, dessen Position im D-Dimensionalen Raum eine potentielle 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 persöhnliches Optimum (lokales Optimum) und hat zudem Kentnnisse über das beste globale Optimum (Bester erreichter Wert der Partikel). 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, übertrumpfen.

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

$$X = [p_1, p_2, \cdots, p_N] \tag{1.6}$$

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

$$p_i = [x_{i1}, x_{i2}, \cdots, x_{iD}] \tag{1.7}$$

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.

$$p_i(t+1) = p_i(t) + v_i(t+1) \tag{1.8}$$

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

$$v_i(t+1) = v(t) + c_1(pb_i - p_i)R_1 + c_2 * (gb - p_i)R_2 \tag{1.9}$$

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 persöhnlichen bzw. globalen Bestwerts gesteuert werden soll. Üblicherweise wird der Bereich von c_1 mit $0 \leq c_1$ 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. Marini und Walczak, 2015). 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 1.9 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 Obaiahnahatti (1998) zeigt, dass das *inherita weight* im Intervall $[0.9, 1.2]$ liegen sollte.

$$v_i(t+1) = \omega(t+1)v(t) + c_1(pb_i - p_i)R_1 + c_2 * (gb - p_i)R_2 \tag{1.10}$$

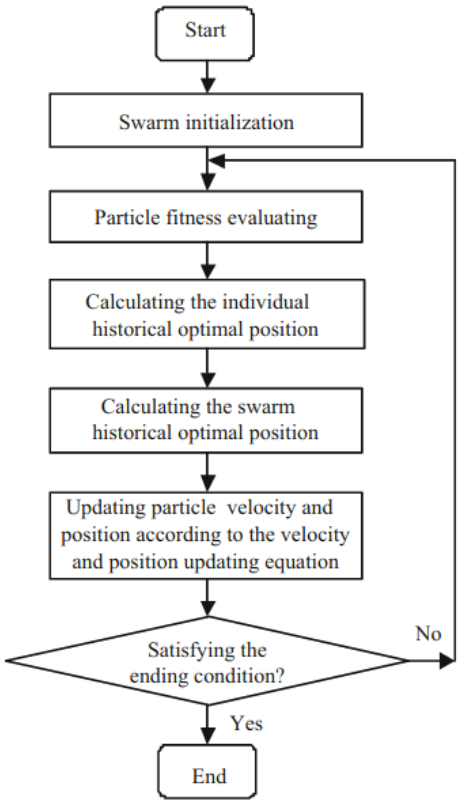


Abbildung 1.4: Flowchart des PSO entnommen aus (Wang et al., 2018)

In Abbildung 1.4 ist der Ablauf des Algorithmus in einem Flowchart dargestellt. Zu Beginn werden die Positionen der Partikel im D-Dimensionalen Suchraum aus einer Gleichverteilung gezogen. Hierbei sind sich viele Authoren einig, dass dies den Suchraum am besten abdeckt. Daraufhin wird der Fehler der Positionen durch die Kostenfunktion berechnet. Dies ist initial der persöhnliche Bestwert jedes Partikels. Zudem wird hieraus der globale Bestwert bestimmt. Es folgt die Berechnung des neuen Richtungsvektors der Partikel $v_i(t+1)$ (vgl. 1.10). Daraufhin erfolgt die Berechnung der neuen Position. Wird ein Abbruchkriterium erreich, wie z.b. das unterschreiten einer Fehlertoleranz oder das Erreichen einer Anzahl an Iterationen, so wird der Algorithmus beendet. Anderenfalls beginnt die nächste Iteration Marini und Walczak, 2015.

2 Stand der Forschung

2.1 Zustände von Schwärmen

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. 1.1.1) folgen zeigen drei unterschiedliche Zustände. Diese werden in dem Papter "Collective States, Multistability and Transitional Behavior in Schooling Fish" von Tunstrøm et al., 2013 als *swarm*, *polarized* und *milling* bezeichnet.

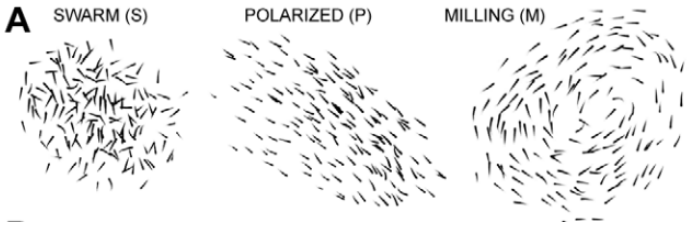


Abbildung 2.1: Zustände von Fischschwärmen , entnommen aus (Tunstrøm et al., 2013)

Swarm bezeichnet globale und lokale unstrukturiertheit. Hierbei ist die Ausrichtung der Agenten scheinbar zufällig. Dies ist in Abbildung 2.1 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 Weiße 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 v_i so kann festgestellt werden ob der Schwarm rotiert. Dies ist der Fall, wenn r_i und v_i im rechten winkel zueinander stehen.

$$O_p = \frac{1}{N}\left|\sum_{i=1}^{N} u_i\right| \tag{2.1a}$$

$$O_r = \frac{1}{N}\left|\sum_{i=1}^{N} u_i \times r_i\right| \tag{2.1b}$$

In obenstehender Gleichung ist die Berechnungsvorschrift für den *Polarized*-Zustand (2.1a) und dem *Milling*-zustand (2.1b) zu sehen.

Eine wichtige Erkenntnis von Tunstrøm et al., 2013 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 $O_r < 0.35$

2.2 Modelle

2.2.1 Metrisches Modell

Wie in Kapitel 1 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 Couzin et al., 2002 in dem Paper "Collective memory and spatial sorting in animal groups" präsentiert. Hierbei wird auf ein Modell gesetzt, das den metrischen Ansatz verfolgt.

Der Wahrnehmungsbereich eines Agenten besteht aus drei Zonen. Die innerste Zone des Models ist die Abstandhalten Zone. In Abbildung 2.2 **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 (2017) ein häufig beobachtetes Phänomen im Tierreich. Die Nachfolgende Gleichung legt das Verhalten des Agenten für diesen Zonenbereich dar.

$$d_r(t+\tau) = -\sum_{j\neq i}^{n_r} \frac{r_{ij}(t)}{|r_{ij}(t)|} \tag{2.2a}$$

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 2.2a beschreibt hier das sich Entfernen des Agenten i von

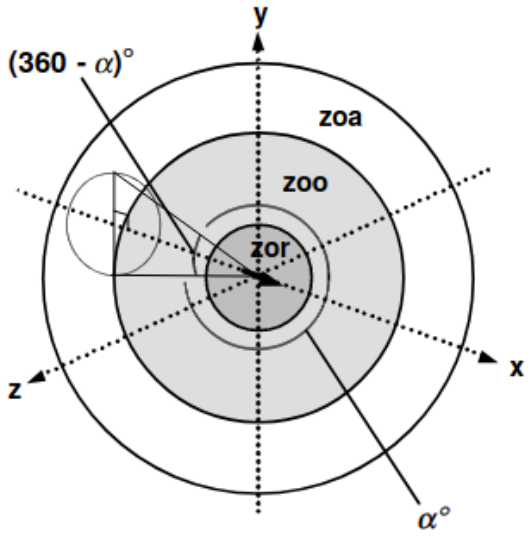


Abbildung 2.2: Zonen im metrischen Modell eines Agenten nach Couzin et al., 2002, Abbildung entnommen aus selbigen Paper

dem lokalen Massezentrum das 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:

$$d_o(t+\tau) = \sum_{j=1}^{n_o} \frac{v_j(t)}{|v_j(t)|} \tag{2.3a}$$

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.

$$d_a(t+\tau) = \sum_{j \neq i}^{n_r} \frac{r_{ij}(t)}{|r_{ij}(t)|} \tag{2.4a}$$

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.

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 entgültige 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 $d_t(t+\tau) = v_i(t)$.

Jeder Agent i besitzt zudem einen blinden Kegel der hinter dem Agenten liegt. Dieser ist in Abbildung 2.2 zu sehen. Der Winkel des Kegels wird durch $360 - \alpha$ berechnet, wobei α als Wahrnehmunsbereich (*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 $d_i(t+\tau)$. Anderenfalls wird die Richtung $d_i(t+\tau)$ des Agenten nicht beschränkt.

2.2.2 Eigenes Modell

Viele Modelle in der gängigen Literatur sind metrische Modelle. Im Kontrast hierzu stehen Modelle, die topolgisch aufgebaut sind. Wie in 1.1.1 besprochen teilen sich die drei Zonen hierbei auf die n-nächsten Nachbarn auf. Ballerini et al. (2008) 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 topolofische Modelle in Angriffsituationen zu bevorzugen, was nicht Teil dieser Arbeit sein wird.

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

Der wichtigste Faktor ist wie bereits erwähnt, das Abstandhalten. 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 wegbewegt, von weiter entfernten Indiviueen genau so stark bestimmt wie von näheren. Jedoch ist es dringender sich dem Agent 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:

$$U_r(t+\tau) = -\frac{r_{ij}(t)}{|r_{ij}(t)|} \cdot f_1(|r_{ij}(t)|) \tag{2.5a}$$

Die funktion f_1 in Gleichung 2.5a 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).

$$U_o(t+\tau) = -\frac{v_j(t)}{|v_j(t)|} \cdot f_2(|r_j(t)|) \tag{2.6a}$$

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 gillt für die nächste Zone.

$$U_a(t+\tau) = \frac{r_j(t) - com(t)}{|r_j(t) - com(t)|} \cdot f_3(|r_j(t) - com(t)|) \tag{2.7a}$$

$$com(t+\tau) = \frac{1}{N}\sum_{i}^{N} c_i(t) \tag{2.7b}$$

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.

In Abbildung 2.3 sind die gewählten Abstandsfunktion zu sehen. Die Faktoren wurde experimentell mit Rücksicht auf den verwendeten Datensatz bestimmt. Ziel ist es drei unterschiedliche Zonen zu erzeugen, durch die die Richtungsvektoren aus den Gleichungen 2.5a,2.6a und 2.7a in Bezug auf die Distanz des dazugehörigen Agenten gewichten.

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

$$d_i(t+\tau) = [\alpha_1 U_a(t+\tau) + \alpha_2 U_a(t+\tau) + \alpha_3 U_a(t+\tau)] \tag{2.8a}$$

$$U_i(t+\tau) = \gamma \frac{d_i(t+\tau)}{|d_i(t+\tau)|} \tag{2.8b}$$

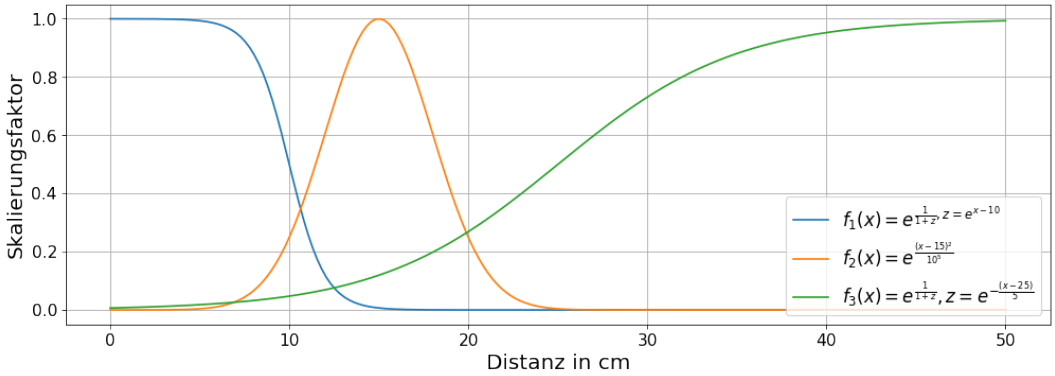


Abbildung 2.3: Skalierungsfunktionen für die verschiedenen Richtungsvektoren

Die Variablen α_i in Gleichung 2.8a 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 2.8b 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. Beispielsweiße 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 2.2.1 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.

2.3 Datensatz

Der für diese Masterarbeit verwendete Datensatz stammt von einem Forscherkollektiv, welches sich auf das Tracking von Tieren anhand von Videos spezialisiert. Jegliche informationen können über deren Homepage www.idtrackerai.io abgerufen werden. Die Tracking Methode wird in dem Artikel "idtracker.ai: tracking all individuals in small or large collectives of unmarked animals" beschrieben Romero-Ferrero et al. (2018). Zu den bereitgestellten Videos, gibt es auf der Homepage zudem die Laufwege der Tiere welche für diese Arbeit verwendet wurden. Es werden die Videos mit 10,60,80 und 100 Zebrafischen verwendet. Die Videos sind alle ca. 30 Minuten lang und mit 32 Bildern pro Sekunde aufgenommen worden.

Im Kern besteht die Architektur zum tracken der Individuen aus zwei künstlichen neuronalen Netzen. In der Vorverarbeitung werden Tiere extrahiert, die entweder zu einzelnen Individuen gehören oder ein Zusammenschluss mehrerer Individuen sind. Ein Netzwerk detektiert daraufhin, ob die extrahierten Tiere einzelne Individuen sind, oder ob ein Zusammenschluss von mehreren Tieren vorliegt.

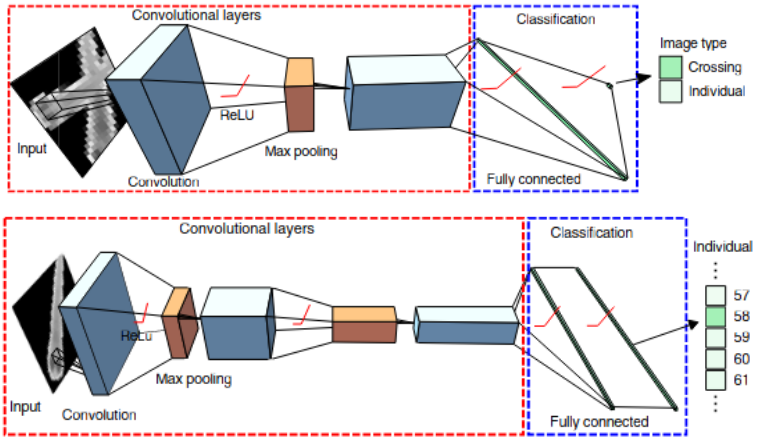


Abbildung 2.4: Netzwerkarchitekturen zur Extrahierung (oben) und Identifikation der Individueen (unten) entnommen aus Romero-Ferrero et al. (2018)

Wie Anhand der Abbildung 2.5 zu erkennen ist, befinden sich die Fische in einem tellerförmigen Behälter.

Abbildung 2.5: Bild aus dem idtrackerai.io projekt. Zusehen sind 60 Fische in einem Tank

3 Experimente

3.1 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 Opimierungsverfahren gegenüberzustellen und herauszufinden, welcher Algorithmus in welchen Situationen zu bevorzugen ist. Folgende Fragestellungen ergeben sich zu diesem Zeitpunkt:

1. Gibt es einen signifikanten Laufzeitunterschied der Algorithmen?
2. Wie gut können die Algorithmen die Parameter Approximieren, wenn diese zwischen den Zeitpunkten konstant sind?
3. Inwieweit verändert sich der Fehler beim Approximieren, wenn die Parameter zwischen den Zeitpunkten schwanken?
4. Wie verändert sich der Fehler, wenn die vorhergesagte Position für den Zeitschritt (t) als Ausgangspunkt für den darauf folgenden Zeitschritt $(t + 1)$ genutzt wird?
5. Lassen sich mit den Modellen die Zustände *Milling*, *Swarm* und *Polarized* erreichen und wenn ja, kann man eine Korrelation der Parameter und den Zuständen beobachten?

Hierzu werden durch die vorgestellten Modellen 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 den Abstand herstellen müssten. Somit würde sich der Zeitpunkt bis ein stabiler Zustand erreicht wird verzögern. Jeder Agent bekommt zu beginn einen Richtungsvektor, der aus einer Gleichverteilung gezogen wird. Dieser Vektor wird hierfür normiert. Dies ist nötig um den Agenten eine Startrichtung zu geben. Wie in Abschnitt 2.2 diskutiert, benötigen Agenten die sich in keiner Nachbarschaft befinden einen Richtungsvektor. Ohne diesen könnte die Richtungsvektoren die sich aus der Orientierungszone ergeben, nicht errechnet werden.

3.1.1 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. Hierbei werden die beiden Algorithmen mit einem initialen Zustand x der Agenten gespeist. Dann werden die Agenten mit einem zufälligen Parameterset an eine Position y manövriert. 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 kleine distanzunterschiede start bestraft, indem der Fehler in die Höhe getrieben wird.

Metrisches Boids Modell

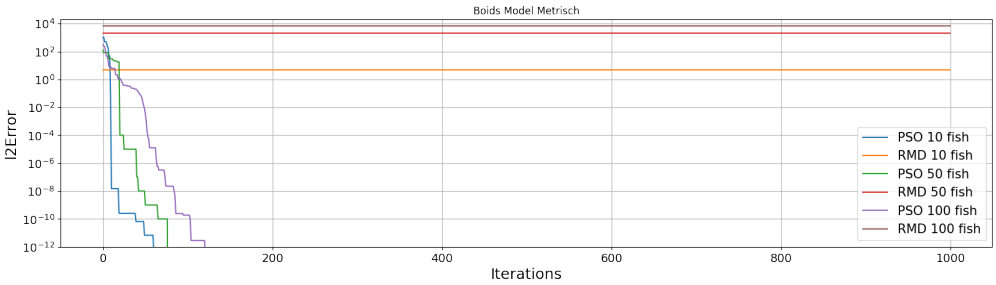


Abbildung 3.1: PSO vs. RMD anhand von Boids

Wie Anhand der Abbildung 3.1 zu erkennen ist, erreicht der PSO Algorithmus im Falle des metrischen Boids modells für alle Agentenanzahlen 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. Das Problem hierbei ist, dass die Parameter nicht Teil des Graphen sind (siehe Abschnitt 1.3.1). Die Parameter für das boids Modell entscheiden nur ob ein Agent zur Berechnung mit einbezogen wird oder nicht. Sie werden nicht bis zur Fehlerfunktion durchgereicht, wordurch die Ableitung bestimmt wird. Alle Versuche die Parameter anderweitig zu verarbeiten scheiterten. Die Laufzeit ist hierbei nicht relevant, da sich der Fehler im Falle des RMD nicht verbessert. Somit ist hier PSO zu bevorzugen.

Eigenes Modell

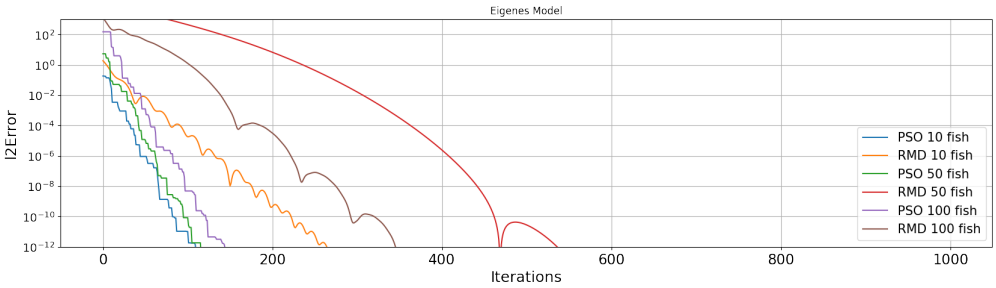


Abbildung 3.2: PSO vs. RMD Anhand des eigenen Modells

Abbildung 3.2 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 3.1: *Vergleich der Laufzeiten zwischen PSO und RMD*

Zu erkennen ist, dass PSO für alle Anzahl an Agenten 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.

3.1.2 Approximierung von konstanten Parametern

Die Approximation der Modellparameter für Schwärme an einem echten Datensatz birgt die Frage wie sich die Parameter über die Zeit verändern und ob dies überhaupt geschieht. Dies ist vermutlich auch Modellabhängig, was allerdings momentan noch nicht betrachtet werden kann. Hier wird eine Simulation betrachtet, die initial Parameter zugewiesen bekommt. Anhand dieser Parameter wird die Simulation über einige Schritte durchgeführt um die Modellparameter daraufhin zu approximieren. Initial werden den Agenten wieder zufällige Positionen und Richtungsvektoren zugewiesen.

Wie zuvor erörtert, ist für dieses Modell nur der 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.

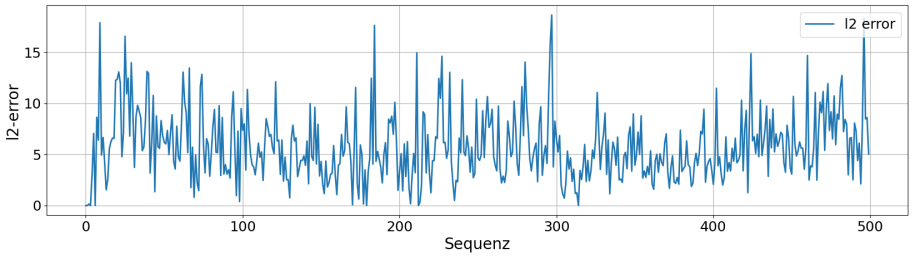


Abbildung 3.3: L2-Fehler der konstanten Parametervorhersage

Der l2-Fehler in Abbildung 3.3 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 der Fehler von Sequenz zu Sequenz größer wird. Die vorhergesagten Agenten driften somit in jedem Zeitschritt von der originalen Simulation ab wodurch sich die Distanzen zwischen den Agenten der Simulation und der Vorhersage vergrößert.

Zudem schwankt der $L2$-Fehler von Sequenz zu Sequenz sehr stark. Dies hat unter anderem damit zu tun, dass der PSO Zufallsbasiert parameter bestimmt. Eine Explosion des Fehlers in dem Sinne, das dieser unweigerlich größer wird, ist hier nicht festzustellen. Wie Anhand der X-Achse zu sehen ist, wurden hier Parameter für 500 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 wurde für die Glättung auf den Wert 6 gesetzt.

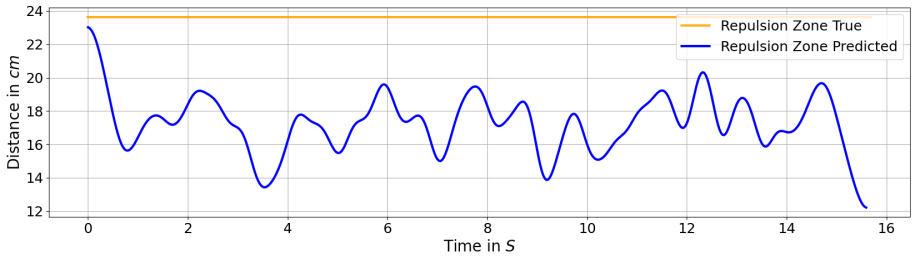


Abbildung 3.4: Vorhergesagte Abstandhalten-Zonen in Blau vs korrekte Zonen in Orange

Die X-Achse spiegelt die Zeit wieder. hierbei wurde angenommen, dass mit 32 Bildern Pro Sekunde simuliert wird. Die Distanz wurde in Zentimeter umgerechnet um mit dem Datensatz vergleichbar zu sein. In den Abbildungen

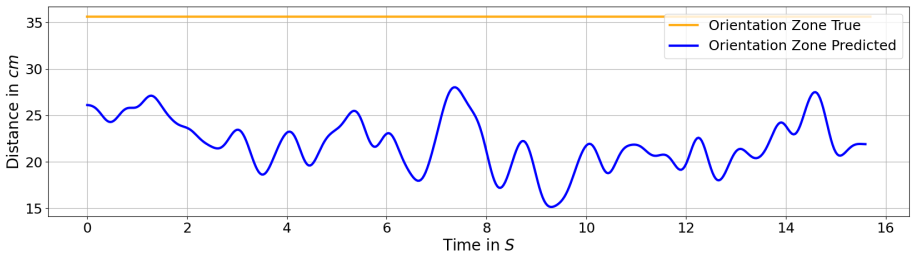


Abbildung 3.5: Vorhergesagte Orientierungs-Zonen in Blau vs korrekte Zonen in Orange

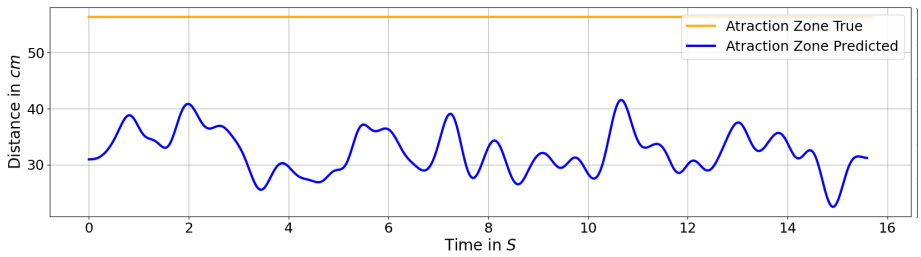


Abbildung 3.6: Vorhergesagte Zusammenhalten-Zonen in Blau vs korrekte Zonen in Orange

3.4-3.6 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 Beispielsweiße ein Agent in der Orientierungszone ist es nicht relevant wie groß die Zone ist, solange nur dieser eine Agent in der Zone vorhanden ist. Selbiges gillt 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.

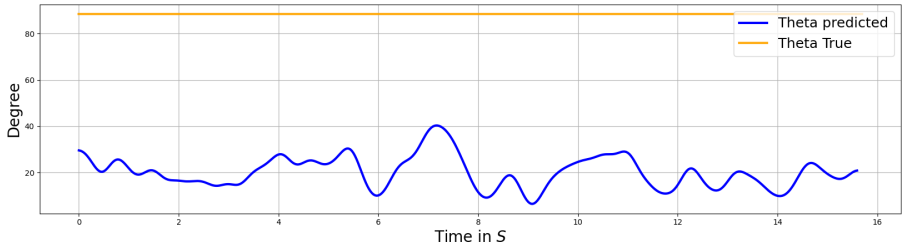


Abbildung 3.7: Vorhergesagter Einschlagswinkel in Blau vs korrekter Winkel in Orange

Der maximale Einschlagswinkel in Abbildung 3.7 wird niedriger vorhergesagt,

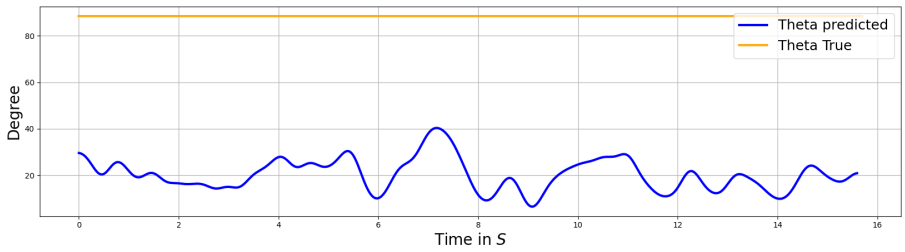


Abbildung 3.8: Vorhergesagte Winkel der blinden Zone in Blau vs korrekter Winkel in Orange

als er tatsächlich ist. Tatsächlich sollte er bei ca. 85 Grad sein, die Vorhersage hingegen schwankt um die 20 Grad. Dies scheint eine hohe Diskrepanz zu sein, jedoch bedeutet ein hoher Einschlagswinkel nicht, dass er auch von den Agenten in der Simulation ausgenutzt bzw. überschritten wird.

Der Winkel θ der den Blindspot definiert wird hingegen als höher vorhergesagt, als er tatsächlich ist. Somit nimmt der Agent durch die 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.

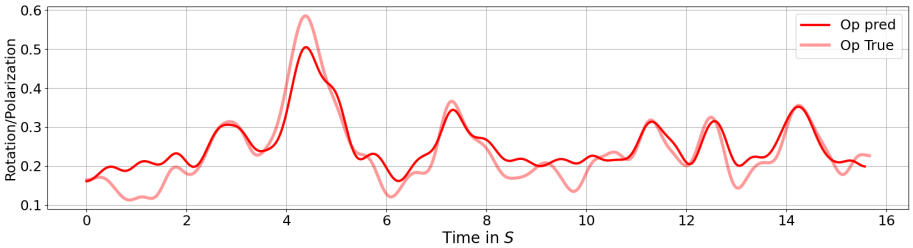


Abbildung 3.9: Polarisierungszustand der Simulation im Vergleich zur Vorhersage

In Abbildung 3.9 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 2.1 besprochen, keine eindeutige Polarisierung vorzufinden ist. Die Kurve liegt in allen Bereichen unter 0.65.

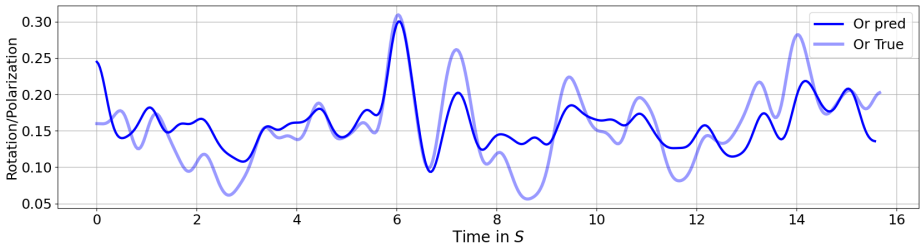


Abbildung 3.10: Rotationszustand der Simulation im Vergleich zur Vorhersage

Die obige Abbildung zeigt ebenfalls, dass die Zustände ähnlich verlaufen. Die Simulation und dessen Vorhersage rotieren demzufolge in ähnlicher Weiße. Auch hier muss erwähnt werden, dass kein Rotationszustand erreicht wurde, da in allen Fällen $O_p < 0.65$ ist. Der Schwarm befindet sich zwischen Sekunde vier und fünf in einem Übergangszustand, denn es gillt $0.35 < O_p < 0.65$ und $O_r < 0.35$. Zu allen anderen Zeitpunkten ist der Schwarmzustand erreicht.

3.1.3 Approximierung von variablen Parametern

3.2 Approximation anhand von realen Daten

Literatur

Ballerini, M., Cabibbo, N., Candelier, R., Cavagna, A., Cisbani, E., Giardina, I., Lecomte, V., Orlandi, A., Parisi, G., Procaccini, A., Viale, M. & Zdravkovic, V. (2008). Interaction Ruling Animal Collective Behaviour Depends on Topological rather than Metric Distance: Evidence from a Field Study. *Proceedings of the National Academy of Sciences of the United States of America*, *105*, 1232–7. https://doi.org/10.1073/pnas.0711437105

Couzin, I. D., Krause, J., James, R., Ruxton, G. D. & Franks, N. R. (2002). Collective memory and spatial sorting in animal groups. *Journal of theoretical biology*, *218 1*, 1–11.

Garland, J., Berdahl, A., Sun, J. & Bollt, E. (2018). The Anatomy of Leadership in Collective Behaviour. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *28*. https://doi.org/10.1063/1.5024395

Hamann, H. (2019). *Schwarmintelligenz*. Springer Spektrum Berlin,Heidelberg.

Majolo, B. & Huang, P. (2017). Group living. https://doi.org/10.1007/978-3-319-47829-6_1865-1

Marini, F. & Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. *Chemometrics and Intelligent Laboratory Systems*, *149*, 153–165. https://doi.org/https://doi.org/10.1016/j.chemolab.2015.08.020

Pankaj K. Agarwal, K. F., Esther Ezra. (2019). Geometric Optimization Revisited. https://doi.org/10.1007/978-3-319-91908-9_5

Reynolds, C. (2002). Steering Behaviors For Autonomous Characters.

Reynolds, C. W. (1987). Flocks, Herds, and Schools: A Distributed Behavioral Model. *SIGGRAPH Computer Graphics*, *21*(4), 25–34. http://doi.acm.org/10.1145/37402.37406

Romero-Ferrero, F., Bergomi, M. G., Hinz, R., Heras, F. J. H. & de Polavieja, G. G. (2018). idtracker.ai: Tracking all individuals in large collectives of unmarked animals. *CoRR*, *abs/1803.04351*. http://arxiv.org/abs/1803.04351

Shi, Y. & Obaiahnahatti, B. (1998). A Modified Particle Swarm Optimizer. *Proceedings of the IEEE Conference on Evolutionary Computation, ICEC*, *6*, 69–73. https://doi.org/10.1109/ICEC.1998.699146

Tunstrøm, K., Katz, Y., Ioannou, C. C., Huepe, C., Lutz, M. J. & Couzin, I. D. (2013). Collective States, Multistability and Transitional Behavior in Schooling Fish. *PLoS Computational Biology*, *9*.

Wang, D., Tan, D. & Liu, L. (2018). Particle swarm optimization algorithm: an overview. *Soft Computing*, *22*. https://doi.org/10.1007/s00500-016-2474-6

Abbildungsverzeichnis

Tabellenverzeichnis