

# Ökosystemsimulation

Jonas Berggren, Jacob Maxton

December 18, 2019

## Abstract

In diesem Dokument erklären wir, wie wir eine numerische Simulation eines Ökosystems entwickelt haben, und was für Erkenntnisse sich daraus ziehen lassen. Die Simulation basiert auf Objektorientierter Programmierung und simuliert die Wechselwirkung zwischen Kaninchen und Fuchsen.

## Contents

<b>1</b>	<b>Die Simulation</b>	<b>2</b>
1.1	Algemeines zu Objektorientierter Programmierung und numerischen Simulationen	2
1.2	Knzept der Simulation . . . . .	2
1.3	Methoden . . . . .	2
1.3.1	Aussuchen eines Ziels . . . . .	2
1.3.2	Bewegung . . . . .	3
1.3.3	Fortpflanzung und Mutation . . . . .	3
1.3.4	Tod . . . . .	3
1.3.5	Andere Methoden . . . . .	3
1.4	Frontend (Jacob) . . . . .	4
1.5	Tarieren der Werte . . . . .	4
<b>2</b>	<b>Die Analyse</b>	<b>4</b>
<b>3</b>	<b>Verbesserungsmoeglichkeiten</b>	<b>4</b>
<b>4</b>	<b>Anwendungen</b>	<b>4</b>
<b>5</b>	<b>Fazit</b>	<b>4</b>

# 1 Die Simulation

## 1.1 Allgemeines zu Objektorientierter Programmierung und numerischen Simulationen

Objektorientierte Programmierung bezeichnet das Programmieren mit Hilfe so genannter Klassen und Objekte. Ein Objekt oder eine Instanz einer Klasse ist beispielsweise ein Individuum des Typs Kaninchen. In dem Beispielwaere die Klasse die kaninchen im allgemeinen. Eine Klasse kann man sich vorstellen wie Methodenkarten in denen eine Reiken von Informationen gespeichert sind. Diese koennen Instanzvariablen oder Methoden sein. Instanzvariablen sind Variablen die jede Instanz traegt. Methoden sind Funktionen oder Anweisungen die jede Instanz der Klasse ausfuehren kann.

Bei numerischen Simulationen wird ein Szenario aus der Reellen Welt durch eine Computer simulation nachgestellt, Dabei wird das System zum Zeitpunkt  $t$  betrachtet. Aufgrundlage dessen wird der Zustand des Systems zum Zeitpunkt  $t + h$  berchnet. dabei gilt je kleiner  $h$  desto genauer ist die simulation, Alle aenderungen werden angewand und der Prozess wird wiederholt bis der gesamte Zeitraum Simuliert wurde den es zu betrachten gilt. Dies ist nuetzlich um die Gultigkeit von Modellen zu pruefen oder z.b. die Stabilitaet des betrachteten Systems zu testen.

## 1.2 Knzept der Simulation

In der Simulation werden Pflanzen Kaninchen und Fuechse simuliert. Jedes Tier ist entweder eine Instanz der Klasse Fox oder Rabbit, die jeweils Unterklassen der Klasse Animal sind. In Animal sind alle Methoden gespeichert, die fuer Kaninchen und Fuechse identisch ausgefuehrt werden, wie der Konstruktor init oder movetargeted zu gezielten bewegen. Methoden die fuer Fuechse und kaninchen unterschiedlcih sind, sind in den Jeweiligen unterklassen gespeichert. Diese sind z.b. findtarget zum finden aller potentiellen Ziele. Auf die unterschiedlcihen Instanzvariablen und Methoden wird aber in Kapitel 1.3 weiter eingegangen. Alle Tiere haben eine kreisfoermiges Sichtfeld, was nicht die komplette Karte abdeckt. Sie haben alle ein, mit der Zeit zunehmendes, Beduerfnis sich fortzupflanzen und zu essen. Alle Beduerfnis werden bei deren Erfuellung verringert. Wird der Hunger zu stark kann das Tier verhungern und sterben. Ausserdem kann ein Tier zu jedem mit einer Wahrscheinlichkeit sterben die vom alter abhaengig ist. Kaninchen essen pflanzen und Fuechse essen Kaninchen, wobei gegessen Kaninche sterben. Kaninchen koennen vor Fuechsen fluechten die sich innerhalb des Sichtfelds befinden.

## 1.3 Methoden

### 1.3.1 Aussuchen eines Ziels

Dies wird durch die Methode findtarget geregelt, die aus der Hauptschleife aufgerufen wird. findtarget ist sowohl eine Methode der Klasse Fox als auch der Klasse Rabbit. Dort sind sie aber unterschiedlich definiert dar fuechse sich anders verhalten sollen als Kaninchen. Jedes Tier speichert alle Objekte die sich innerhalb des Sichtradius self.sens befinden in einer Liste ab.

Anschliessend wird geprueft welches Ziel angesteuert werden soll. Dazu wird aus jeder Liste das naechste Element gesucht. Wenn ein Kaninchen ein Fuchs sieht hat die Flucht immer oberste Prioritaet. Der Fuchs wir anwisiert un der Bewegungsvektor wird mit -1 multipliziert danach wird

gepueft ob Hunger oder Libido staerker wirkt und demnach wird entschieden ob das naechste essen oder naechste partner anwisiert wird.

### 1.3.2 Bewegung

Die Bewegung der Tiere wird durch die Methoden `movetargeted` und `moverandom` definiert. Jedes Tier hat zwei Arten wie es sich fortbewegen kann. Wenn es ein bestimmtes Ziel hat, wird `movetargeted` aufgerufen, und das Tier bewegt sich entlang des Vektors von der eigenen Position zum Ziel. Dabei ist der Bewegungsvektor auf die Bewegungsgeschwindigkeit normiert. Wenn kein Ziel in Sicht ist, wird `moverandom` aufgerufen, und sie bewegen sich zufällig. Dabei wird die Methode `Collision` jedes mal aufgerufen. Diese verhindert das Tiere aus der Katze rauslaufen.

### 1.3.3 Fortpflanzung und Mutation

Haben sich zwei gefunden wird der Liste der Tiere eine neue Instanz der Klasse hinzugefügt. Bei den mutierbaren Eigenschaften wird der Durchschnitt aus den jeweiligen Werten der Eltern als Mittelwert angenommen. Der Wert des Kindes weicht um  $x$  von diesen Durchschnitt ab, wobei  $x$  eine zufällige Zahl zwischen  $u$  und  $v$  ist

Die Eltern speichern sich gegenseitig als ehemalige Partner ab und betrachten sich anschließend für eine festgelegte Frist nicht mehr als mögliche Partner. Ausserdem koennen sich die Eltern prinzipiell fuer eine festgelegte Frist nicht weiter fortpflanzen.

### 1.3.4 Tod

Tiere koennen auf drei unterschiedlichen Arten sterben. Sie koennen verhungern, sie koennen durch die Altersabhaeuge Funktion sterben oder Kaninchen koennen verhungern. Dies wird durch die Methoden `starve`, die `eat` geregelt. Die Methoden `stave` und die werden bei jeder Iteration aufgerufen, und `eat` wenn ein Kaninchen gefressen wird.

Die Todeswahrscheinlichkeitsdichte wird durch eine Funktion beschrieben nach dem Muster:

$$a(e^{-t+b} \cdot c \cdot t + d) \tag{1}$$

$t$  entspricht dem Alter fuer  $t \in \mathbb{R}_{>0}$  und  $a$  bis  $d$  sind Konstanten. Durch so eine Funktion ich  $P$  fuer ein kleines  $t$  relativ gross.  $P$  erreicht im positiven Bereich ein Minimum und wächst dann approximativ linear an. Dies fasst Kindersterblichkeit und altersschwäche in eine Funktion zusammen.

Beim sterben durch gegessen werden löscht der fuchs die gegessene instanz aus der Liste der Tiere.

### 1.3.5 Andere Methoden

Hinzu kommen noch weitere Hilfsmethoden wie z.B. `distance` die den Abstand zwischen `self` und einem beliebigen Punkt berechnet. Diese sind jedoch fuer das allgemeine Verständnis unsere Arbeit nicht essentiell.

## 1.4 Frontend (Jacob)

## 1.5 Tarieren der Werte

# 2 Die Analyse

# 3 Verbesserungsmoeglichkeiten

Eine Moeglichkeit das Programm zu verbessern waere ein sogenanntes Gridsystem zu implementieren. Dabei wird das Feld in mehrere Subfelder unterteilt. Die Objekte werden in einer Matrix gespeichert wobei jeder Index der Matrix eine Subfeld zugeordnet ist. Das bietet den Vorteil, dass die Tiere bei Jeder Iteration nicht mehr die Die position aller Objekte abfragen muessen, sondern nur die der Objekte die sich im selben bzw. in einem der Felder befinden die mit dem Schtkreis ueberlappen. Somit kann die laufzeit pro Iteration stark reduziert werden.

Asserdem kann eine andere, schnellere Programmiersprache verwendet werden. Fuer Unfanreiche wissenschaftliche Anwendungen kann eine erweiterte Version eines solchen Programm auf staerkeren Computern ausgefuehrt werden.

# 4 Anwendungen

Eine Umfanreichere Simulation dieser Art koennte nuetzlichsein um die Auswirkungen von Menschlichem Eingriff in oekosysteme wie Klimaerwaermung, Lebensraum veraendungen oder austerben einer Art abzuschuetzen. Dies koennte nuetzlich sein um notwendigkeit und Dringlichkeit von Umwelt Massnahmem abzuschuetzen.

# 5 Fazit

Wir haben mit Hilfe objektientierter Programmierung eine Vereinfachte Simulation eines OEkosysteme entwickelt im dem wir uns auf die Wechselwirkung zwischen zwei Tierarten konzentriert haben. Dabei sind wir auf unterschiedliche Schwierigkeiten gestossen. zuerst mussten wir entscheiden wie genau unsere Simulation sein soll. Anschliesend war es sehr schwierig all Willkuerrlich gewaehlten Parameter so zu beziffern, dass keine der Arten zu schnell ausstrirbt. Dabei haben wir festgestellt wie fragil auch so ein einfaches OEkosystem sein kann.