FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

# Meistern von unterschiedlichen Computerspielen mittels Generation Based Learning

BACHELORARBEIT

zur Erlangung des akademischen Grades

**Bachelor of Science**

im Rahmen des Studiums

**Medizinische Informatik**

eingereicht von

**Manuel Esberger**
Matrikelnummer 01525631

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Univ.Prof. Dr. Allan Hanbury
Mitwirkung: Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle

Wien, 30. September 2018

| Manuel Esberger | Allan Hanbury |
|---|---|

FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

# Mastering Divserse Computer Games using Generation Based Learning

BACHELOR'S THESIS

submitted in partial fulfillment of the requirements for the degree of

**Bachelor of Science**

in

**Medical Informatics**

by

**Manuel Esberger**
Registration Number 01525631

to the Faculty of Informatics

at the TU Wien

Advisor: Univ.Prof. Dr. Allan Hanbury
Assistance: Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle
Pretitle Forename Surname, Posttitle

Vienna, 30th September, 2018

Manuel Esberger

Allan Hanbury

# Erklärung zur Verfassung der Arbeit

Manuel Esberger
Preßgasse 11/2 1040 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 30. September 2018

Manuel Esberger

# Danksagung

Zu erst möchte ich meinen Großeltern danken. Sie haben mir angeboten bei ihnen zu wohnen als ich mein Studium angetretten habe. Ich werde mich vermutlich mein Leben lang daran erinnern, dass ich bis spät in die Nacht in die Tastatur hämmerte um Tätigkeiten zu erfüllen, die das Studium von mir verlangten, während mein Großvater versuchte im Nebenzimmer Schlaf zu finden. Er hat es sich nie anmerken lassen, dass mein Lernen nicht nur mich wachgehalten hat. Auch erinnere ich mich an die vielen grantigen Diskussionen mit meiner Großmutter wenn es im Studium mal nicht so rund lief. Auch sie hat mir jedes zornige Wort verziehen und sie grüßt mich weiterhin Willkommen in ihrem Heim. Hätten sie mir nicht ihre Türen offen gehalten und mir einen Platz zum Lernen angeboten, wäre das Studium vermutlich nicht möglich gewesen.

Weiteres möchte ich meiner Freundin und baldigen Mutter meines Sohnes Danken. Auch wenn es abseits vom Studium viel zu Tun gab, wie zum Beispiel Möbel kaufen, dem Job oder den nicht enden wollenden Arztbesuchen, erinnerte sie mich immer wieder daran an meiner Bachelorarbeit zu schreiben. Auch wenn Sie manchmal fragte ob wir etwas Zeit für uns haben wollen und ich sie auf Grund der Arbeit abwieß, zeigte Sie sich mit Verständnis.

Zu guter Letzt möchte ich auch allen Professorinnen, Professoren und Universitätsangestellten danken, die nicht nur an der Wissensvermittlung interessiert waren, sondern die auch aktive Schritte gesetzt haben um interessierten Studierenden bei ihrem Lernprozessen zu unterstützen. Ich denke die Universität würde nicht ohne ihnen funktionieren und ich habe sie auch für mein Studium schätzen gelernt.
Vielen Dank!

# Acknowledgements

Enter your text here.

# Kurzfassung

Ihr Text hier.

# Abstract

Enter your text here.

# Contents

CHAPTER 1

# Introduction

*"Some people worry that artificial intelligence will make us feel inferior, but then, anybody in his right mind should have an inferiority complex every time he looks at a flower."*

— Alan Kay, *(Computer Scientist)*

https://sokogskriv.no/en/writing/structure/structuring-a-thesis/ http://www.charleslipson.com/How-to-write-a-thesis.htm

## 1.1 Motivation and Problem Statement

In the last decade may different solutions for neuronal networks (NN) have been implemented, whereas these implementations propose various changes like the amount and distribution of connections between neurons, the weight calculations between neuronal connections or the amount of neuronal layers of the network as well as other structural decisions. The efficiency of these algorithms depend on the problem space, which they were tested on. For example ....

xor problem

concrete examples

1. einleitung komplexe aufgaben
2. bisher: nn networkd
3. genetic algorithms
4. nn + ga => neat and others
5. `https://www.reddit.com/r/NeuralNetwork/comments/4nea5i/how_to_decide_what_neural_network_architecture_to`
6. One complex problem are diverse games.

7. different games and criteria
8. different algorithms and neat
9. expectations of this bachelor work

## 1.2 Results

1. what was interesting to see
2. contrast to expectations

### 1.2.1 Some References

## 1.3 Thesis Structure

**Chapter 2**

**Chapter 3**

**Chapter 4**

**Chapter 5**

CHAPTER 2

# Related Work

## 2.1 Genetics Algorithms, genetic programs

https://www.quora.com/Whats-the-difference-between-Genetic-Algorithms-and-Genetic-Programming http://outlace.com/miniga.html https://stackoverflow.com/questions/1402370/when-should-i-use-genetic-algorithms-as-opposed-to-neural-networks

## 2.2 Artificial Neuronal Networks

https://stackoverflow.com/questions/1402370/when-should-i-use-genetic-algorithms-as-opposed-to-neural-networks

## 2.3 NEAT

https://stackoverflow.com/questions/45390481/what-is-neat-neuroevolution-of-augmenting-topologies

`https://www.reddit.com/r/NeuralNetwork/comments/3a1zjh/some_basic_questions_about_implementing_neat/`

## 2.4 Tools

1. MarI/0
2. Flappy Bird
    - NEAT Flappy
    - Machine Flappy

3. Python for statistics

CHAPTER 3

# Generation Learning in Computer Games

1. What was measured: fitness development within neats generations
2. two different games: marI/O and flappy
3. different challenges within the game
4.

## 3.1 MarI/O

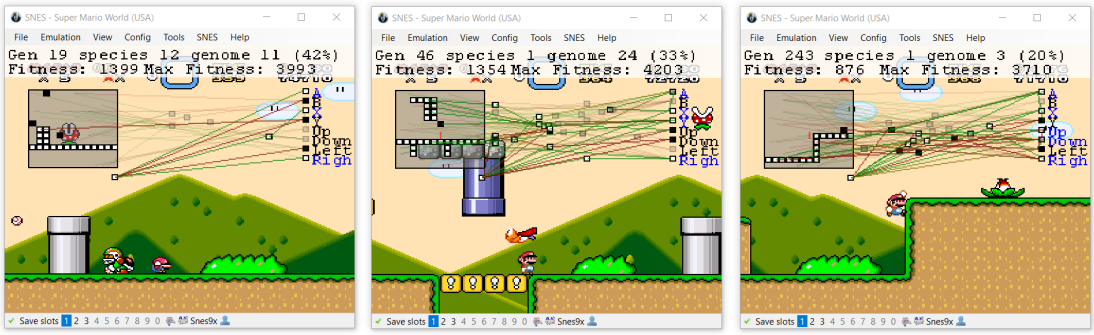


Figure 3.1: MarI/O simulation

As mentioned in section 2.4 MarI/O (see figure 3.1) is an implementation of Neuro-Evolution of Augmenting Topologies (NEAT)-algorithm written in LUA. It provides a solution for automatic learning of the game Super Mario World. In Super Mario World a

level is a two dimensional map with steady as well as moving obstacles. Some of them block the path to the goal and others cause damage to Mario's health. A few of them give health upgrades to Mario or add coins to the players account, although the coins are ignored in the implementation of the Artificial Intelligence (AI). Since there are many different positions in which Mario can stay and the speed of the game depends mostly of the player and his/her/its desicions the environment of Super Mario World is rather complex when compared to the second game Flappy Bird 3.2.
This complex world leaves the expectation that many hundred thousand runs are necessary to learn how to complete a level. However MarI/O implementation reached to goal after approximately 2664.29 runs on average in the simulations described later in this section. Still 2 of the 9 simulations didn't reach the goal once.

fitnessfunction, formlar? when was goal reached

Later in this section 3 figures with 3 homomorphic graphs will be shown. The three different figures display the success of the algorithm in different classes of initial population size. Since the NEAT-algorithm used does not produce a deterministic amount of populations after the first generation (in general: Generation 0), the initial population size define these classes. There where three classes choosen with a scaling factor of 5 between them. These initial population sizes are 10, 50 and 250. Whether or not the initial population sizes are well chosen will be discussed shortly in the conclusion section (see 3.3) of this chapter. Depending on the evolution of the NN, there are a certain amount of generations evolved. Every generation contains their own set of species. And on the other hand the species contain the genomes. In generation 0 every species contains only one genome each. The sum of all genomes in all species of a generation is called the population. In the cases where the initial population size is 10 or 50 over time to many generations where created to show a viewable graph in the end. That's why only 30 generations where picked in the display with even distances between them. Still a continuous line with the best run of a population is showed above all generations, even the skipped ones.
In the later descriptions of the population classes there are two types of runs introduced. First is the "plot-run" which indicates the simulation and the graph. Inside of this graph there were many "runs" which represent the runs of the population (genomes) of each species. In figure 3.1 there are 3 individual runs displayed. On average one plot-run consists of $4217.\overline{2}$ runs, whereas population 10 has 2828 runs on average, population 50 contains of $4494.\overline{6}$ runs on average and population 250 of 5329 runs averagely.
In order to understand the fundamental differences of these simulations, the population classes are examined in more detail:

**Population 10 / Generation 500** As it is visible in figure 3.2 the vertical axis shows the fitness score average of the genomes within a species. The horizontal axis portrays the generations containing the species. Each generation contains up to 10 populations which is divided into species and genomes within species. This species devision was made based on the NEAT algorithm described in section 2.3. The best run of the genomes grouped by each generation is marked with a blue line. Therefore the blue line

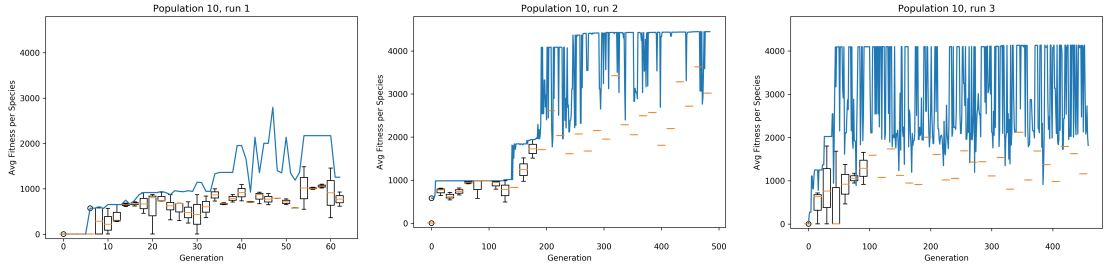


Figure 3.2: MarI/O Population 10

indicates the best overall run within a generation. Since the boxplot portrays the species's avarage score of each generation and the the blue line shows the best run per genome (population), the boxplot and the blue line rarely meet. Still the average population score is closer to the best run than in the next two population variants (see later in this section population 50 3.1 and population 250 3.1). This can be calculated by taking the median of the species fitnesses and substracting that number from the best run of the genomes: $average\_distance = \frac{\sum_{g_i \in generations} max(g_i.genomes) - median(g_i.species)}{|generations|} \approx 1107$ whereas $g_i.genomes$ and $g_i.species$ are lists of the respective fitnesses.
In the three plot-runs on average $334.\overline{6}$ generations were created, which results in a skipping of generations inside the graphics of around $11.1\overline{5}$ generations averagely, between two displayed generations. Unfortunately the first run crashed after generation 60. Still, because of the long runtime of the simulation the plot-run was kept. However, indicated by plot-run 2 and 3, the population growth started after this generation. As it can be seen in the 3rd plot-run of the figure 3.2, sometimes runs over 3000 fitness score could be achieved even after the 30th generation. In plot-run 2 the average fitness of the single species left tend to rise, however more and longer plot-runs would be needed to test this hypothesis.
In each generations there are up to 10 populations. In the first generation (Gen 0) no mating was done. So in the first generation there where 10 species spawned with one genome each. In the 10th Generation on average only $4.\overline{3}$ species where left. After generation 50 maximum 3 species where left in all runs and after generation 190 in plot-run 2 and after generation 91 in plot-run 3, respectively, only 1 species was left for mating. The mating results into the corssover of species.

(up to because of neat implementation check neat implementation

All runs except plot-run 1 reached the goal (the end of the level) multiple times which can be seen by the fitnesscore being over 4000. However plot run 3 reached the goal the earliest with runs over 4096 starting from generation 44. Still there was the most overall regress made in plot run 3. This can be calculated by adding the differences between the best runs of each generation if the difference was negative: $average\_regress = \frac{\sum_{g_i \in generations} min(max(g_i.genomes) - max(g_{i-1}.genomes), 0)}{|generations|} \approx -348$ again whereas $g_i.genomes$ is a list of the fitnesses of each genome inside the generation. The regress of plot-run 1 was $-88.27$ approximately and of plot-run 2 was around $-109.98$.

In plot-run 1 the $average\_fitness\_increase = \frac{\sum_{g_i \in generations} max(g_i.genomes) - max(g_{i-1}.genomes)}{|generations|} \approx 19.87$ was the biggest of the three plot-runs since the first plot-run ended early and plot-run 3 had many drawbacks. The average fitness increase of plot-run 2 was around 7.97 and of plot-run 3 was only 3.95 approximately. Since it is only slightly possible to extend the maximum score above the score of 4000 and plot-run 1 has never reached this ranking, plot-run 1 pointed out to have the best score increase per round. Every successful round, whereas Mario reached the goal will only minimize the fitness increase when averaged with the generation count. In another words, for an infinitely large amount of generations the $average\_fitness\_increase$ is expected to converge to 0 since the game has and end-state in contrast to the game Flappy Bird, as it can be seen in section 3.2. In mathematical terms: $\lim_{n \to \infty} average\_fitness\_increase(n) = 0$, whereas the $average\_fitness\_increase(n)$ is defined as the average fitness increase of a set of n generations.

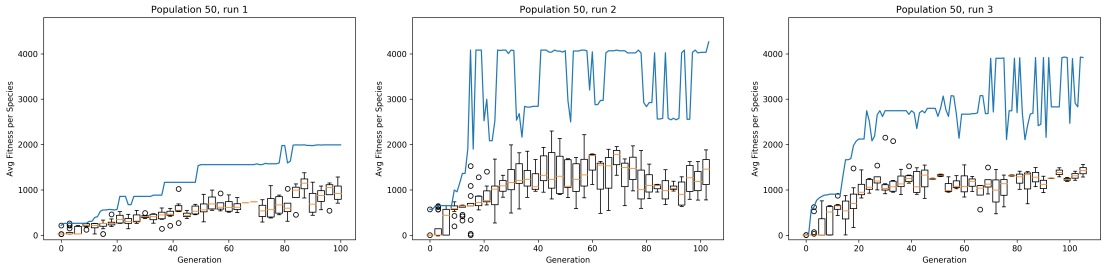


Figure 3.3: MarI/O Population 50

**Population 50 / Generation 100** In this setup the population count is up to 50, again distributed into species and genomes within species according to the MarI/O NEAT implementation. The $average\_distance$ between the median of the species of each generation to the best genome run of this generation is bigger than that of the simulation with it's initial population size of 10 but it is smaller than in the last case. The $average\_distance$ was calculated as described in the previous simulation (population 10 3.1) and the value is approximately 1406.
In this simulations the plot-runs where executed until there where over 100 generations (101 generations in plot-run 1, 104 in plot-run 2 and 106 in plot-run 3). This results in an average skipping of 3.456 generations between the display of two generations.
In generation 0 there where 50 species spawned, again, with one genome each. In the 10th generation there where 15 species left on average. At the end of generation 100 on average $3.\overline{3}$ species where left from the initial 50 generations.
Interestingly the plot-run 1 couldn't learn to reach the goal. From this data it is not trivial to predict if the breakthrough would have started within the next 50 generations or if this plot-run would have stayed low in it's fitness score, since there are no clear patterns to find in the graphical representation of these runs. In order to answer on this question more profoundly, further and longer plot-runs have to be made and the big

jumps between the fitness scores of each neighbour generation would have to be analysed. Plot-run 2 and 3 had more luck in reaching the end, however plot-run 3 had more stability in it's high score results between generations. Still after generation 70 plot-run 3 also shows stronger differences between it's generation's heigh scores. Nevertheless, plot-run 2 reached the goal the earliest. The first time plot-run 2 achieved a fitness-score over 4000 was in generation 15 (it reached a score of 4082.5), whereas plot-run 3 reached a maximum score of 3928 in generation 98. Still plot-un 3 reached to goal with a score of 3902 the first time in generation 70. The 3rd plot-run has the highest $average\_fitness\_increase \approx 36.92$ of the three plot-runs. Plot-run 1 has an $average\_fitness\_increase$ of 17.58 approximately and plot-run 2 of 35.5 precisely.
Plot-run 2 and 3 have similar $average\_regress$ values with approximately $-158.02$ for plot-run 2 and $-152.37$ for plot-run 3. Because of the early end of general low performance of plot-run 1 the $average\_regress$ is also the lowest with $-6.30$. Still there where only 15 cases where the succeeding generation performed worse than the previews in plot-run 1 whereas there where 29 of these cases in plot-run 2 and 30 in plot-run 3. This indicated a certain stability in the first plot-run although the maximum score remained far lower than 3000.

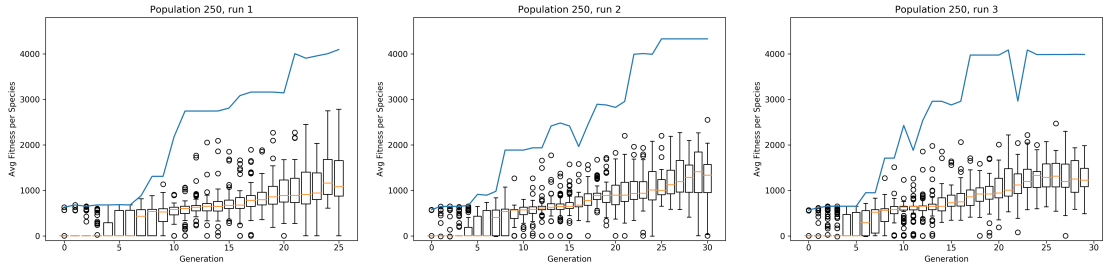


Figure 3.4: MarI/O Population 250

**Population 250 / Generation 30** In figure 3.4 the population size is up to 250 in generation 0. In the first generation (Gen 0) 250 species are born with one genome each. The $average\_distance$ for this plot-runs is the biggest with approximately 1827 when compared to the plot-runs with an initial population size of 10 and 50. In this plots no generations had to be skipped in order to portray a descriptive graph since the maximum generation count is 30 in plot-run 2 (25 generations in run 1 and 29 generations in run 3). Already in the 6th generation, on average only 94.8 species where left. At the end of generation 25 there where 31 species left on average.
Compared to the other two population classes there are at least 7 times more species left at the end of the simulations which results in longer whiskers of the boxplot.The wiskers even contains bad starts with fitness-scores lower than 100 in plot-run 1 and 2. Interestingly the best runs are always exceptions after generation 6 (in plot-run 1 and 2 even earlier).
Further it is to mention that the plots are rather uniform compared to the plots of population 10 and 50. Therefore the $average\_fitness\_increase$ has similar values with

a low variance which are 133.25 for generation 1, around 121.08 for generation 2 and 113.95 for generation 3. The *average_regress* is the lowest in plot-run 1 with $-5.87$ approximately. This is because the maximum value of the succeeding generation is smaller then the previous generation in only 4 cases. The other two plot-runs have an *average_regress* of approximately $-19.97$ in plot-run 2 and $-61.92$ in plot-run 3. All of the plot-runs reached the end of the level even thought plot-run 3 reached the end at generation 17, whereas plot-run 1 reached the end at generation 23 and plot-run 2 at generation 22.

| MarI/O | avg. runs /$\sigma$ | avg. fitness score /$\sigma$ | avg distance /$\sigma$ | avg. regress /$\sigma$ | avg. fitness increase /$\sigma$ |
|---|---|---|---|---|---|
| Population 10 | 2828 /2055.44 | 1231.42 /531.37 | 1107.09 /534.5 | -182.03 /144.01 | 10.6 /8.28 |
| Population 50 | 4494.$\overline{6}$ /176.09 | 960.96 /321.34 | 1405.96 /664.75 | -105.56 /86.01 | 30 /10.78 |
| Population 250 | 5329 /656.74 | 776.31 /57.88 | 1826.32 /81.79 | -29.25 /29.16 | 122.76 /9.76 |

Table 3.1: MarI/O Population Comparison Overview

Differences and similarities between runs in graph

**Comparison of the results**

In order to compare the results, 5 distinct values (see table 3.1) of the plot-runs where calculated, as three of them were introduced in more detail earlier in this section 3.1. The first observations indicate that the average fitness score of each generation drops when establishing a bigger initial population. However the standard deviation tend to drop as well.

Also the distance of the median of the species to the best run of the generation seam to become greater with a greater population count in generation 0. However, the average regress (if present) becomes lower with bigger population sizes and fewer generations, as well as it's deviation. Since there are fewer generations in the simulations with an initial population of 250 and these simulations having similar achievements, the average fitness increase is higher than in the other two simulation classes. The standard deviation of the average fitness increase is relatively similar.

It is interesting to see how the fitness increase compares to the average distance value. Even thought the fitness increase of population class 250 is much higher than the fitness increase of population class 10, the distance remains largs which indicates that the majority of runs stayed low and the average score of population class 10 is higher than in the other two population classes. Still the other two classes remained more stable when taking the average regress into account.

Another interesting point of view is the reaching of the end of the level (the goal). In all population classes the goal was reached even thought population class 1 and 2 didn't reach the goal in one plot-run each. Population 10 reached the goal in the first 24.48% on average, only including the cases where the goal was reached. Population 50 reached the goal in the first $40,24\%$ on average and population class 250 in the first $69,47\%$.

To summarize the results roughly it can be said that an initial population size of 10 promises faster and better results in an complex environment like Super Mario World, however more stability can be reached when increasing the initial population size.

## 3.2 Machine Learning Flappy Bird

change this title

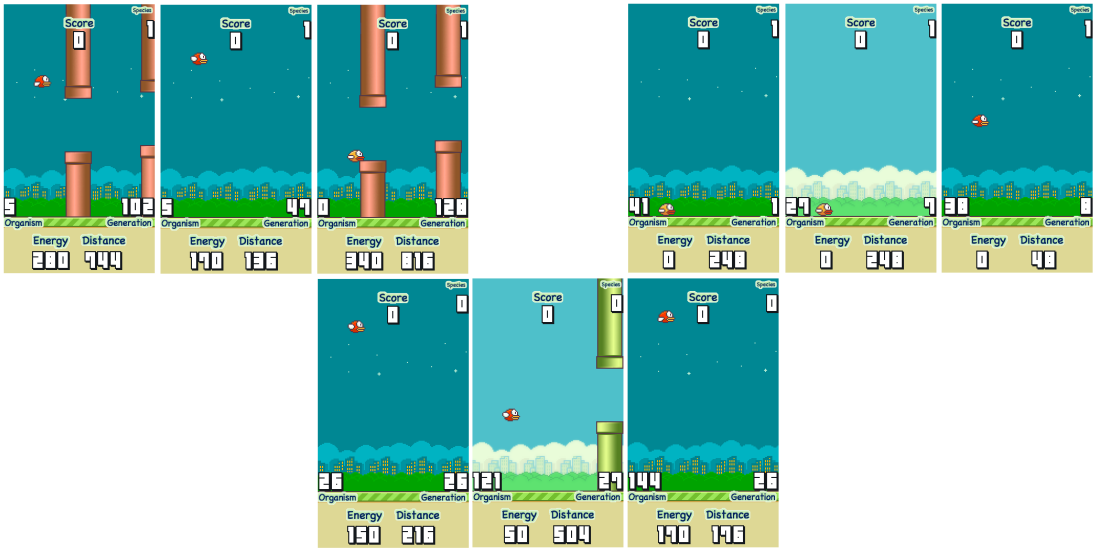


Figure 3.5: Flappy Birds simulation

1. explanation of environment and expectations
2. fitnessfunction, formlar?
3. explanation of graph of population (10, 50, 250) averaged on generations (30 generations evenly choosen [equal spaces between generation numbers]) (abstract explanation)
4. check if expectations of marI/O can confirm
5. huge difference between best runs and majority of runs (extreme luck), => double graph
6. Differences between runs
7. unexpectedly bad results
8. => Neat vs other machine learning
    a) Differences between runs (lucky runs with 4th champion generation)

**Population 10 / Generation 500**

**Population 50 / Generation 100** a

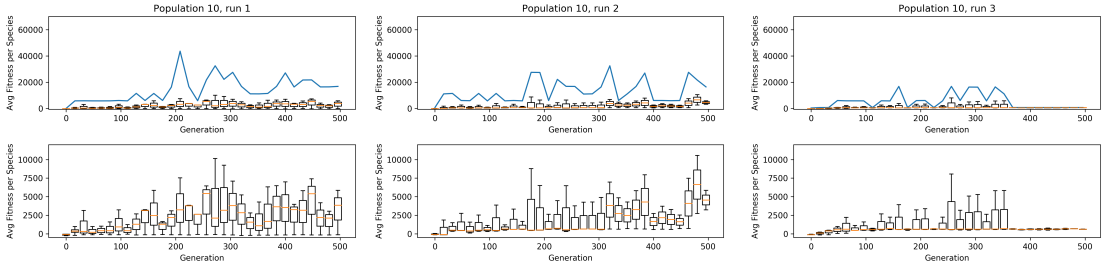


Figure 3.6: Flappy Bird Population 10

**Population 250 / Generation 30**

1. best runs are exceptions (outside of whiskers)

### 3.2.1 Plain Machine learning flappy bird

1. better results
2. multi simulation made it easier
3. easy algorithm for easy environment might be explanation for better results

## 3.3 Conclusion

1. Population sizes smart? (10, 50, 250)
2. differences / similarities in implementation (fixed size in machine learning flappy bird whereas dynamic species with marI/O)
3. differences / similarities in outcome
4. future studies
   - genome/generation plot & differences to other plot
   - compare calculations with all data not only with maxfitness in case of *average_fitness_increase* and *average_regress*
   - check in text for (future or further)
5. test MarI/O previous evolutions on other levels (short 5)
6. table with all the data collected

CHAPTER 4

# Comparison and Meta-Analysis

## 4.1 Concepts Section 1

## 4.2 Parameters

1. abstract parameters
    - population size (maybe not choosen well => futer work)
2. many wheels that can be turend
3. differen setup => similar goal
4. further work, checking influence of nn parameters

## 4.3 Conclusion

CHAPTER 5

# Conclusion

*"By far, the greatest danger of Artificial Intelligence is that people conclude too early that they understand it."*

— Eliezer S. Yudkowsky, *(Artificial Intelligence Researcher)*

https://sokogskriv.no/en/writing/structure/structuring-a-thesis/ http://www.charleslipson.com/How-to-write-a-thesis.htm

- 
- test MarI/O previous evolutions on other levels (long: what would be expected, how long to adapt to differences, would be better than start from anew)

## 5.1 Stuff

## 5.2 Future Work

CHAPTER 6

# Additional Chapter

Enter your text here.

# List of Figures

# List of Tables

# List of Algorithms

# Glossary

**LUA** "Lua is a powerful, efficient, lightweight, embeddable scripting language. It supports procedural programming, object-oriented programming, functional programming, data-driven programming, and data description."[1]. 5

[1] https://www.lua.org/about.html

# Acronyms

**NEAT** Neuro-Evolution of Augmenting Topologies. 5

**NN** neuronal networks. 1

# Bibliography


[App10a] Apple Inc. *Keynote '09 User Guide.* Apple Inc., 2010.

[App10b] Apple Inc. *Numbers '09 User Guide.* Apple Inc., 2010.

[App10c] Apple Inc. *Pages '09 User Guide.* Apple Inc., 2010.

[Fre10] Free Software Foundation, Inc. Gnu general public license, 2010.

[Jür95] Manuela Jürgens. *LaTeX: Fortgeschrittene Anwendungen.* FernUniversität Gesamthochschule in Hagen, 1995.

[Jür00] Manuela Jürgens. *LaTeX: eine Einführung und ein bisschen mehr.* FernUniversität Gesamthochschule in Hagen, 2000.

[KJUM11] Markus Kohm and Jens-Uwe-Morawski. *KOMA-Script: Die Anleitung.* 2011.

[Mie11a] André Miede. *A Classic Thesis Style: An Homage to The Elements of Typographic Style*, 2011.

[Mie11b] André Miede. A classic thesis style by andré miede, 2011.