Malte F. Speidel

Path Influenced Environments and Navigation Approaches Within

FAKULTÄT FÜR
INFORMATIK

Intelligent Cooperative Systems
Computational Intelligence

Path Influenced Environments and Navigation Approaches Within

Bachelor Thesis

Malte F. Speidel

November 30, 2023

Supervisor: Prof. Dr.-Ing. habil. Sanaz Mostaghim

Advisor: M. Sc. Julia Heise

Advisor: Dr.-Ing. Jens Weise

Malte F. Speidel: *Path Influenced Environments and Navigation Approaches Within*
Otto-von-Guericke-Universität
Intelligent Cooperative Systems
Computational Intelligence
Magdeburg, 2023.

Abstract

In this Bachelors Thesis, we will explore the topic of Path-Influenced Environments. We will discuss those environments, also known as PIE's, and how they differ from normal pathfinding environments. As Path-Influenced Environments are currently not extensively researched, we will also examine the performance of widely utilized pathfinding algorithms such as A-Star and an evolutionary algorithm in those environments. With that, we try to answer if those approaches are suitable to navigate through those environments, how they compare to each other using metrics like the length of the path and how much energy they used. We also want to determine which modifications are necessary to make these approaches viable in this new kind of environment. To answer these questions, we execute the EA and the A-Star algorithm on four different maps and look at the results.

Preface

First and foremost, I want to thank my friends and family for the never-ending support. I also want to thank all members of the Swarm Lab at Otto-von-Guericke-University since the regular meetings never failed to provide useful tips and inspiration while writing this thesis. A special thank-you goes to my advisors Jens Weise and Julia Heise for putting up with the stress of reading this thesis over and over again and being there when any questions arose.

Contents

List of Figures **VI**

1 Introduction **3**

2 Related Work **7**

2.1 Pathfinding 7
2.2 A-Star 7
2.3 Optimization 8
2.4 Evolutionary Algorithm 8
2.5 Epistasis 9
2.6 Multimodality 10
2.7 Fréchet Distance 10
2.8 Manhattan Distance 11

3 Methods **13**

3.1 Problem Definition 13
3.2 Algorithms and Modifications 15
3.3 Implementation 17

4 Experiments **21**

4.1 Experiment Goals 21
4.2 Evaluation Criteria and Research Questions 22
4.3 Experiment Setup 22
4.3.1 Map 1 24
4.3.2 Map 2 25
4.3.3 Map 3 26
4.3.4 Map 4 27

5 Results and Evaluation **29**
5.1 Map 1 29
5.2 Map 2 31
5.3 Map 3 32
5.4 Map 4 34
5.5 Figures 35

6 Conclusion **47**

7 Future Work **49**

Bibliography **50**

List of Figures

2.1 Evolutionary Algorithm 9

3.1 Combination of Objects 14

4.1 Map 1 — Random Pattern 24
4.2 Map 2 — Valley Pattern 25
4.3 Map 3 — Checkerboard Pattern 26
4.4 Map 4 — Random Pattern with horizontal Goal 27

5.1 Map1 Run 1 Fitness Progression 35
5.2 Map 1 Run 1 Final Path 36
5.3 Map 1 Fitness Distribution 36
5.4 Map 1 Fitness Value Box Plot 37
5.5 Map 1 A-Star Path 37
5.6 Map 1 Run 1 and Run 4 38
5.7 Map 2 Run 18 Fitness Progression 38
5.8 Map 2 Run 18 Final Path 39
5.9 Map 2 Fitness Distribution 39
5.10 Map 2 Fitness Value Box Plot 40
5.11 Map 2 A-Star Final Path 40
5.12 Map 3 Run 1 EA Final Path 41
5.13 Map 3 Run 1 Fitness Progression 41
5.14 Map 3 Fitness Distribution 42
5.15 Map 3 Fitness Value Box Plot 42
5.16 Map 3 A-Star Final Path 43

5.17 Map 4 Run 1 EA Final Path 43
5.18 Map 4 Run 1 Fitness Progression 44
5.19 Map 4 Fitness Distribution 44
5.20 Map 4 Fitness Value Box Plot 45
5.21 Map 4 A-Star Final Path 45
5.22 Map 4 Run 1 and Run 5 46

1 Introduction

As described in an article by Stout [11], the typical pathfinding problem consists of obstacle avoidance. We would rather not avoid obstacles here, but are instead eager to work with them. As mentioned in an article by Algfoor, Sunar and Kolivand [13], pathfinding is used in many day to day things, such as GPS and video games. These applications consider many things, such as traffic and other goals, but what is never changed is the structure of the map itself. This is very reasonable, since we have no possibility to move a street or walkway to suit our purpose and get a more efficient path to the goal. For some environments, however, we have to consider the possibility, that we can move obstacles around. Taking that concept a bit further, we can ask what if we can not only move objects around, but combine them and basically stack them together? A snowplow would be an example of that. While it moves through the snow, the snowplow pushes it aside and onto other snow and therefore combines the two small piles, making them heavier and harder to move. Those environments, in which we can not only move obstacles around, but can also combine them, are called "Path Influenced Environments". Normally, environments just contain either walkable space or immovable objects. Here we are not only dealing with those two kinds of obstacles, but also with movable objects, that can be pushed around and combined with each other to form a more massive object. This creates an ever-changing environment, which is harder to navigate. Looking into the real world, we can spot some problems that fit our definition. We already discussed the snowplow as a possible example. With it, we change the environment, which will have an impact on further navigation and movement. Another example for those environments is digging a tunnel. During excavation, certain actions may

prevent us from proceeding in a particular direction due to stability concerns or excessive density of the material. The result of that is an environment that changes based on the path we take, which is the definition of a "Path-Influenced Environment". Since we now have explained what a PIE is and have shown that there are real-world examples, the next thing we have to look at is why we should research pathfinding in this "new" type of environment. The simple answer is that, in the spirit of the academic discipline, we should find ways to navigate them, but another and more general answer to this question comes if we look at the real-world examples. If we want to automate snowplowing, for example, we need to know efficient ways to navigate the vehicles through the environment. Furthermore, if we take a look at the environmental impact, we want to complete this task as energy efficient as possible. The same goes for the excavation of tunnels and other problems that might fit our definition.

Something that we should discuss beforehand is what approaches for the navigation of PIE's already exist. Since this is a relatively new field of research, at the time that this thesis is written, there is no proven best way to solve those problems. In light of this lack of reliable solutions, we will take a look at how algorithms, that would typically be used for related problems, could be modified to fit our problem and how well they perform on it.

With that in mind, we will answer the following research questions in this thesis:

1. Can an evolutionary algorithm navigate PIE's?
2. Can A* navigate PIE's?
3. How efficient is the EA in comparison to A*?
4. Is there multimodality between paths?
5. Is there epistasis?

To answer those questions, we will perform runs of A* and an evolutionary algorithm in multiple PIE's and collect metrics that will help us determine the performance of both. This will also allow us to compare them to each other and possibly determine which approach is more suitable.

As for why we use A-Star to navigate these environments, in many papers and articles such as an article by Candra, Budiman and Pohan [4], an article by Barnouti, Al-Dabbagh, Naser [1] and others, this algorithm was used successfully and since A-Star is very stable and deterministic in nature, it is seemingly very fitting to use this algorithm here.

As for the structure of this thesis, we will start with the chapter "Related Work" in which we will explain the technical terms used here. Following that, we proceed to the chapter "Methods". This includes the problem definition, algorithms that are used here and how they were implemented. We will furthermore take a look at how we modified these algorithms to make them fit the problem. After finishing this, we will proceed with the chapter "Experiments". In this chapter, we will clarify what settings we use for the experiments. This will answer the following questions: "What do we want to evaluate?", "What results do we expect?", "What experiments do we want to perform?", "Which metrics do we use?" and most important, "How do these experiments give an answer to our research question?". When we finished evaluating the experiment setup and answering the listed questions, we will follow up with the chapter "Results and Evaluation". This chapter includes the execution of the experiments, evaluation of the results and a discussion of the results with focus on the proposed research questions. The subsequent chapter will be the conclusion. This will summarize the points mentioned earlier and will also aim to make a final statement regarding the research question. At the end of this thesis, we will discuss future work and how the research field of this thesis could be expanded. Now that we have a general overview about what is coming next, we can proceed with the next chapter, "Related Work".

2 Related Work

The following section will provide definitions for terms that will be used throughout the remainder of the thesis and are essential for further comprehension, such as “Pathfinding”, “A-Star Algorithm” and others.

2.1 Pathfinding

The initial term we shall examine is the term “Pathfinding”. In the Introduction of Cui and Shi’s paper [6], pathfinding is described as finding the shortest route between two end points.

2.2 A-Star

Referring again to Cui and Shis’s paper [6], A-Star is a generic pathfinding/search algorithm. This algorithm begins with a set start and end tile/-position. We take the start tile and look at the immediate neighbors. If the neighbors are walkable, which means that they are neither a wall nor a solid object, a so-called “heuristic cost” is determined. This heuristic cost is calculated as the sum of the distance to the start and the estimated distance to the end tile. The neighbors with their respective heuristic costs are then put into a queue. This queue will always be sorted in a way, so that the tile with the lowest heuristic cost is at the front. We then take this tile and if it is the end tile, the algorithm is finished, otherwise the same process repeats itself. It is important to mention that, if we look at a tile that is already in the queue, we

compare the old heuristic value with the new one. If the new one is lower than the old one, the old value will be overwritten. Pseudocode for this algorithm can be seen in the chapter Methods.

2.3 Optimization

In the Book "Handbook of Research on Modern Optimization Algorithms and Applications in Engineering and Economics" authored by Vasant, Weber and Dieu [10], optimization is defined as finding the best available values of a given function or domain. What we want to optimize here is the fitness value. The fitness value is an indicator of how well an individual performs its task. The higher the value, the better. Since we use an evolutionary algorithm, we select a group of the "fittest" individuals and use them to create a new population. We repeat this step for each generation and therefore optimize the fitness value by selecting the best ones available.

2.4 Evolutionary Algorithm

In the abstract of Bartz-Beielstein, Branke, Mehnen and Mersmann's paper [2], the term evolutionary algorithm is described as an umbrella term for "population-based stochastic direct search algorithms" that try to emulate natural evolution. In our case, we generate a population of individuals and test how well they perform in the given environment. The best performing individuals are then selected, and a new population is created from their attributes and structure. In general, an EA starts with a random initial population. Each individual in this population has a mostly unique set of genes. In this context, genes represent the encoded actions the individual wants to take. With the initial population, we simulate every individual in the environment, that we want to test them in. This results in a so-called fitness value, that is calculated based on how well the individual performed. When this is done, we take the best individuals and generate a new population based on them. This new population

of individuals is then tested again, and this cycle repeats until the maximum number of generations is reached. This number has been set beforehand. In the next figure, we can see this procedure in a diagram.

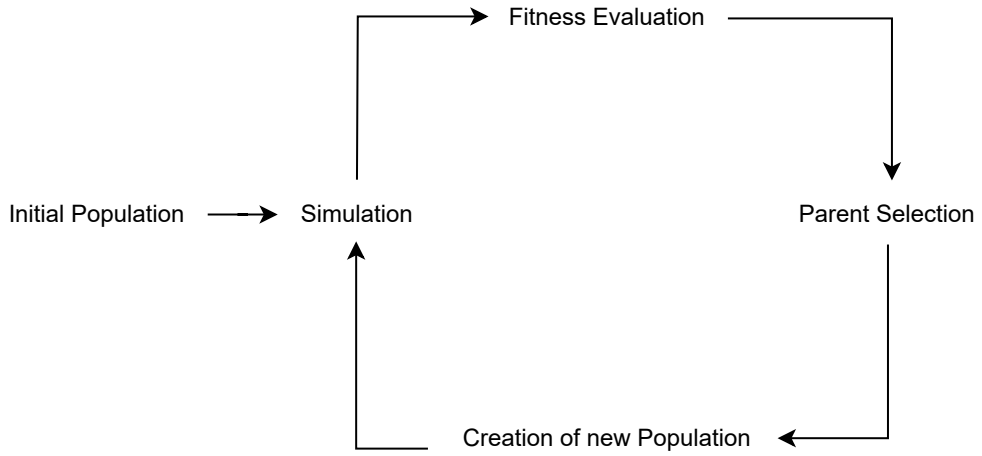

Figure 2.1: Evolutionary Algorithm

2.5 Epistasis

According to Cordell's Paper [5], the term "epistatic" was first used by Bateseon in 1909 in "Mendel's Principles of Heredity" [3]. It was used to describe that one variant on a locus prevents the variant on another locus to manifest its effect. This principle is applicable to an evolutionary algorithm. If we take the position of an agent and let it step to the right and immediately back to the left, the second move prevented the first one from having any effect, and therefore we have a case of epistasis. How this exactly works in our environment will be specified in the chapter "Results and Evaluation".

2.6 Multimodality

In the book written by Kalyanmoy Deb [7], multimodality is defined as having multiple optimal solutions, of which many are local optimal solutions. In the context of our problem, we might have multiple paths to the goal that are the same regarding length and energy used to complete them. We are only interested in the best solutions, so what we want to know is if the best value of the EA or the A-Star algorithm for that matter, can be reached multiple times with different paths.

2.7 Fréchet Distance

The Fréchet Distance itself was originally defined by M. Fréchet [8]. In Weise and Mostaghim's paper [12], the Fréchet Distance is explained as a similarity measure for curves in a metric space. Imagine having two curves in the same metric space. The Fréchet Distance would be the maximum distance between points of the curve. Mathematically speaking, the formula would be the following.

$$\delta_F(A, B) = \inf_{\alpha,\beta} \max_{t\in[0,1]} \{d(A(\alpha(t)), B(\beta(t)))\}$$

With the Fréchet Distance, we can compare paths to each other and determine how similar they are. This is helpful, when we want to compare the solutions from A-Star and the evolutionary algorithm to each other and if we want to determine if we have multimodality. To be more specific, we use the discrete Fréchet distance here. It is necessary to mention that the Fréchet distance also has its drawbacks. When we have a path that has a distance of one unit to another path, we do not get information about how long we held a distance of one unit between the paths. This means the path could always have a distance of one to the other, or just for a short section. As a basic similarity measure, this should suffice.

2.8 Manhattan Distance

According to M. D. Malkauthekar's paper [9], the Manhattan Distance is defined as follows:

$$MH(a,b) = |x_1 - y_2| + |x_2 - y_2| + ... + |x_n - y_n| = \sum_{i=1}^{n} |x_n - y_n|$$

This distance measure is relevant for this thesis, since the Manhattan Distance is used in both A* and the evolutionary algorithm. The exact explanation where it is used and why can be found in the chapter methods.

3 Methods

In this chapter, we will go over the problem definition itself and the used methods. We will also take a brief look at how things are implemented.

3.1 Problem Definition

The first step in defining the problem is to examine the surrounding environment. This environment consists of three types of obstacles. We have pockets of air, which have no resistance and no energy requirement to move over or into them. The second type is movable obstacles. This type requires energy to be moved. It can also be combined with other obstacles, which increases their density and therefore the energy required to move the new object. The third type is the type that we know from other pathfinding problems, an immovable object. This kind of object cannot be moved at all, and also cannot be combined with other movable objects. In this environment, we use a four neighborhood, which means that from each position that the agent is at, it can move into four directions. Now that we know the structure of the environment, we have to clarify two special interactions with the movable objects. Firstly, it is not possible to move an object through the agent. This means that a move, where the agent would switch places with a movable object, would be illegal. Secondly, we cannot decide where to move the object itself. We cannot say that we want to move the object up, down, left or right. The direction in which the object will be moved will be the direction that poses the least resistance. An example of this can be seen in the figure below.

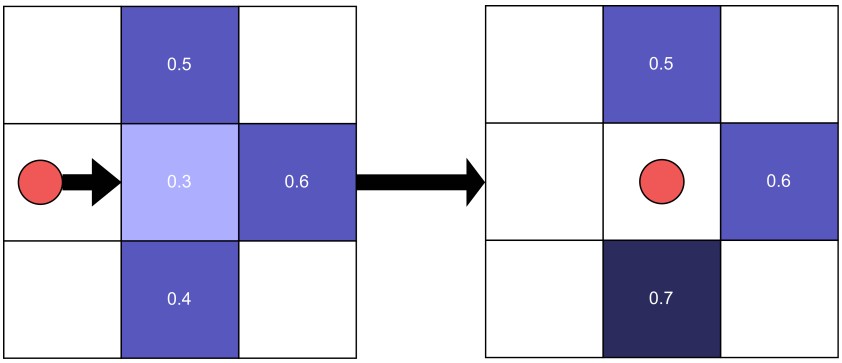

Figure 3.1: Combination of Objects

So let's suppose that the agent approaches from the left and tries to move the object to the right. Since we use a four-neighborhood, the object would have four ways in total to move. The direction where the agent pushes the object from can be ignored, since swapping places with the obstacle would be an illegal move. From the now three remaining directions, the one with the least "heavy" or "dense" object is chosen and is then combined with the object we want to move. One thing that has to be added here is that the density after the combination is also a deciding factor for the combination of both objects. If the new object would exceed a set max density, then the objects cannot be combined. In general, we want to have a reliable way to navigate those environments. For this purpose, we want to use a modified A-Star algorithm, since it performs very well on known pathfinding problems and thus will act as our baseline for later comparisons. We also want to use an EA, since the randomness in its evolutionary steps will, eventually, find a suitable solution to the problem. In combination with these two approaches to the problem, we want to examine if we can observe epistasis in the evolutionary algorithm. This means that we want to see if some paths will not be able to manifest if another path has already manifested. We also want to know if we can detect cases of multimodality. We might also be able to compare the two approaches to each other and decide which method is more suitable.

3.2 Algorithms and Modifications

We already mentioned that we are using a modified version of the A*-Algorithm. A normal A*-Algorithm would begin knowing the start tile and the end tile that it wants to visit. At the start tile we would look at the neighbors of it and determine if they are "walkable", so if they are neither a wall nor another immovable object. If they are walkable, we calculate three values for them. The first one is the distance to the start α. The second one is the estimated distance to the goal β. To estimate the distance to the goal, we use the "Manhattan Distance". The third value that we need is the heuristic value or heuristic cost γ. The heuristic cost tells us how much we probably have to invest to reach the goal from that position. This leaves us with the following formula for the heuristic cost: $\gamma = \alpha + \beta$. After we got the three values for each of the walkable tiles, we put them into a list or queue. From this list, we then grab the tile with the lowest heuristic cost and repeat the same process until we find the end tile. That was the algorithm in its normal form, and now we have to look at the modifications we employed to fit it to the problem. We mainly did two things. The first one is, that we expanded the "walkable" criteria. We not only test if we would be walking into a wall, out of bounds or into an immovable object, but we also test if there is even the possibility for the moved object to advance to another position. As discussed in the problem definition, there could be a wall next to the object with which it cannot be combined with, or the neighboring objects could be too dense to form a new one. If those tests succeed, we can calculate the heuristic value for the new tiles, and here we have the second modification. The distance to the start α and the estimated distance to the end tile β are as described above, but the heuristic cost γ is now calculated as follows:

$$\gamma = 0.5 \cdot (\alpha + \beta) + 0.5 \cdot \text{Used Energy}$$

Since we want to be as energy efficient as possible, while we are moving through the environment, we factor in the energy that we used to move to this place. With those two modifications, the algorithm should be suitable to

solve the problem. Now onto the evolutionary algorithm. A EA always has a population of individuals. Those individuals consist of genes and a fitness value. In our case, the genes look like this: "[SOUTH, WEST, WEST, NORTH, EAST, ...]". We encode the moves that the individual takes, beginning from the starting position, with the corresponding cardinal directions. At the start of the EA, a population of individuals is randomly generated. Here we generate 20 individuals, each with 20 genes. This population size was chosen to make sure that we have enough individuals to get diverse solutions to the problem. Those individuals are evaluated, and a fitness value is calculated. We then take four of the best individuals and mutate them. Those mutations can vary between adding a new direction to the genes and removing or modifying an existing direction. In a state-of-the-art EA, we would also do something that is called "Crossover". This means that the genes of two individuals are split at one or multiple points and then recombined with each other. We do not do this here, since the modifications made by the random mutations should suffice. After a new population is generated, the process repeats itself until the number of generations passed meets the upper boundary, which has to be specified beforehand. To ensure that we get a good result, the maximum number of generations is set to 3000. This number seems very high, and this is the case. If we were interested in any solution that the EA can come up with, it would suffice to set the maximum number of generations to 100, for example. But since we want to guarantee getting as close to a perfect solution as possible, the number is set much higher. To fit the EA to our problem, we did a few things. To obtain the energy used by the individual, each individual is simulated on the map we want to test it on. When everything is simulated, we can calculate the fitness value of each individual. We start with a base value of 100. From that, we deduct a weighted sum of the number of steps taken and the energy used. The energy requirement for each obstacle stems from its density multiplied by the factor ten, DENSITY $\cdot$ 10 = ENERGY USED. The exact formula looks as follows:

$$\text{FITNESS} = 100 - (0.01 \cdot \text{NUMBER OF STEPS} + 0.01 \cdot \text{ENERGY USED})$$

3.3 Implementation

In this section, we will list the pseudocode for the A-Star and the evolutionary algorithm. Note that these implementations are not state-of-the-art, and they contain the modifications that we discussed beforehand.

Algorithm 1 A-Star Algorithm

```
Require: startNode, endNode
  openList ← []                                          ▷ Empty open list
  closedList ← []                                        ▷ Empty closed list
  openList ← startNode                                   ▷ Enqueue start node
  while openList ≠ empty do
    currentNode ← openList[lowestHeuristicCost]
    closedList ← currentNode
    if currentNode = endNode then
      Finished
    end if
    neighbors ← neighbors of currentNode that are walkable and ∉ closedList
    for all neighbors do
      if neighbor ∈ openList then
        if heuristic cost neighbor < heuristic cost openListNeighbor then
          Swap openListNeighbor with new neighbor
        end if
      else
        openList ← neighbor
      end if
    end for
  end while
```

Algorithm 2 Evolutionary Algorithm

```
procedure MUTATION
    if geneLength > 1 then
        numberOfMutations ← randomNumber ≤ geneLength
    end if
    a ← randomNumber, 0 < randomNumber < 100
    if a < 33.33 then
        remove random gene
    else if a < 66.66  &  a > 33.33 then
        add random gene
    else
        Change random gene
    end if
end procedure

procedure GENERATEPOPULATION(bestIndividuals)
Require: numberOfChildren
    newPopulation ← bestIndividuals
    for all individuals ∈ bestIndividuals do
        i = 0
        while i < numberOfChildren do
            newIndividual ← oldIndividual.mutate()
            newPopulation ← newIndividual
            i++
        end while
    end for
return newPopulation
end procedure
```

Require: population, numberOfGenerations, genCounter
  while genCounter $<$ numberOfGenerations **do**
    for all individual $\in$ population **do**
      simulate individual ▷ let individual run on map
      $Fitness \leftarrow 100 - 0.01 \cdot stepsTaken - 0.01 \cdot energyUsed$
    end for
    Sort population for best fitness values
    bestIndividuals $\leftarrow$ population[0...3]
    population = generatePopulation(bestIndividuals)
    genCounter++
  end while

4 Experiments

In this chapter, we will go over the setup of the experiments and determine what we want to evaluate. We will also examine the outcomes that can be expected from the experiments, and also go over the metrics that will be employed.

4.1 Experiment Goals

As stated in the Introduction and in the chapter "Methods", for the EA, we want to evaluate the energy requirements and the length of the generated paths. Moreover, we desire to determine whether the algorithms are capable of identifying a path to the goal and, specifically for the EA, to determine the number of generations it took to discover it. Regarding the genome for the individuals of the evolutionary algorithm, we will determine if epistasis and multimodality can be detected. Since A-Star has a deterministic path, meaning that the path the algorithm takes will not change as long as the map stays the same, we are going to use the paths that A-Star generates as ground truth and measure how similar the EA paths are to it, using the Fréchet distance. We will also take a look at the respective fitness scores and how they evolve. Here it might be beneficial to mention that we will also calculate the fitness values for the A-Star paths in the same fashion as we do for the EA. The algorithm normally does not have a fitness value for its paths, but since the calculation is based on energy used and steps taken, we can calculate the respective values and use those as another criteria for comparison.

4.2 Evaluation Criteria and Research Questions

The first two questions, asking if the A-Star algorithm and the EA are capable of traversing PIEs, are the easiest to answer, being either affirmative or negative. If we see that the respective approach reached the goal, the answer is clear. In contrast to the EA, A-Star will only be executed, since we already mentioned that the outcome of this algorithm is deterministic. The third question was how efficient the EA is in comparison to A-Star. Our metrics make it possible to find multiple answers to this question. We can look at the number of steps taken, and the energy used separately, or we can use the calculated fitness values to determine the difference. Out of these options, we select the length of the path and the fitness value. As for the fourth question about the existence of multimodality, we will examine paths that have the same fitness value and use the Fréchet distance to determine how similar they are. With that, we will also be able to determine how much the paths diverge for the respective maps, provided that we have multimodality. For the last question of epistasis, we can take a look at different goal paths of the EA. If we were to compare them based on their genome, it would be plausible to argue that a given sequence of actions and the resulting map states, one path cannot lead to the evolution of the other path.

4.3 Experiment Setup

Generally, we have 4 Maps with different structures and challenges. On each map, the evolutionary algorithm will be executed 31 times with 3000 generations per run. This comparatively high number of generations can be explained by the fact, that we do not use a state-of-the-art EA. As stated previously, we omit the crossover, for example, which makes the EA not as performant as it could be. With the high number of generations we want to increase the opportunity to get close to a result, that is near the perfect solution or we at least can get over local optima. For the starting conditions, we have 20 individuals per generation and start with a random genome of length of 20. A-Star will only be run once

per map due to its deterministic nature. As for the maps, the thickness of each obstacle was determined randomly. The size of each map is 20×20 or 22×22 if we want to consider the walls that are on the edges. The start for each map is at the coordinates $x = 1, y = 1$. In the following, we will go over each of the 4 Maps individually and what results we expect on each of them.

4.3.1 Map 1

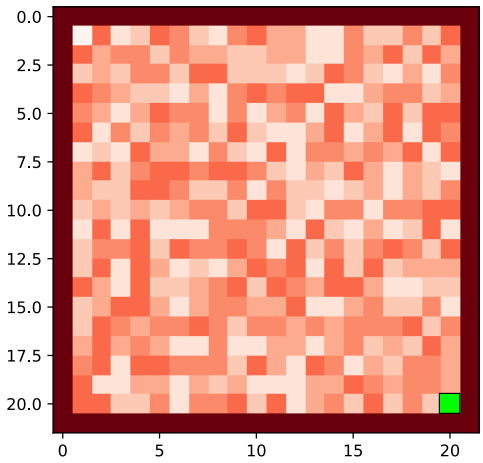

Figure 4.1: Map 1 — Random Pattern

The first map contains a random pattern. Obstacles were created randomly at will, without a specific path or solution for this environment in mind. Their density was selected pseudo randomly by the python math module. The end point of this map is at $x = 20, y = 20$, which is the furthest point from the start. We can expect that both algorithms solve this map, since there are no obvious traps here, and the density of the obstacles is low enough so that we cannot trap ourselves. As for the length of the path and the energy required, we cannot reliably predict an outcome here, since we have no method of knowing what the best route is. The same argument is applicable to the question of multimodality, but we can expect that there are goal paths with the same fitness value since we have numerous possible paths available.

4.3.2 Map 2

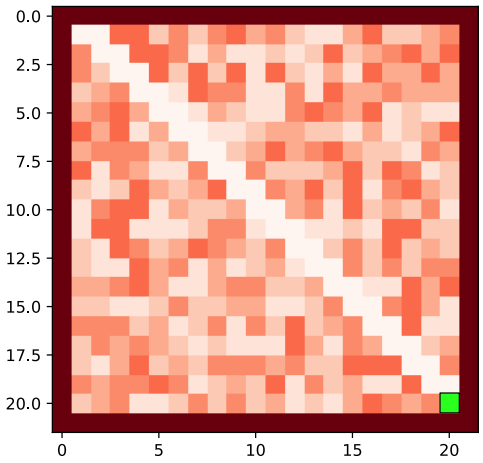

Figure 4.2: Map 2 — Valley Pattern

The second map contains a well-known pattern, the valley pattern. This map has a path of air pockets directly to the goal. What we want to observe here is if the approaches can take the best way to the goal. On other maps, we cannot know without a very deep search what the best path is. As of writing this thesis, there appears to be no algorithmic approach to this, and testing every path on a 20 by 20 map with 4 directions for each step is exhausting. These obstacles are overcome by simply creating a path that is both the shortest and the most energy-efficient path, with a length of 38 and an energy requirement of 0. As for our expectations, this is the easiest map to predict. On one hand, A-Star should create a path with 38 steps through the diagonally placed pockets of air, and the energy consumption should be 0. On the other hand, the EA might perform worse than A-Star since its moves and therefore the end path are random. It is true that we should see improvement over an increasing number of generations due to the survival of the fittest, but the best result of a generation might not be the optimal result for this path, resulting in us straying away from the optimal path.

4.3.3 Map 3

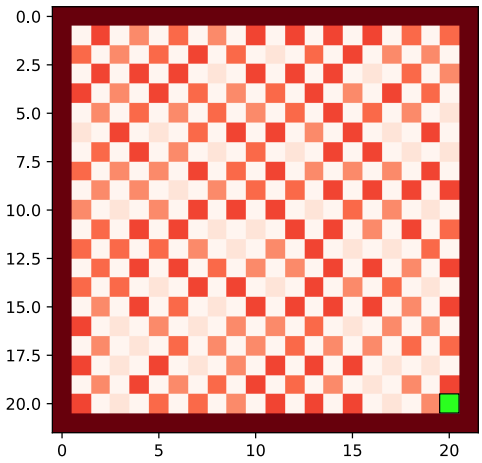

Figure 4.3: Map 3 — Checkerboard Pattern

The third and map contains a checkerboard pattern. Here, we are generally interested in how each approach solves the problem. Through the structure of the map we can generally refrain from combining obstacles since we always have the possibility of moving the obstacle into a pocket of air, but not combining obstacles may lead to a longer path. A-Star should be able to find the path for this pattern. It always had a pocket of air with 0 density available to move an obstacle into. In contrast to that, the EA might perform worse on its first goal path since it can create scenarios with its random moves that lead to combining obstacles and therefore a higher energy consumption, but this should improve drastically with the further generations.

4.3.4 Map 4

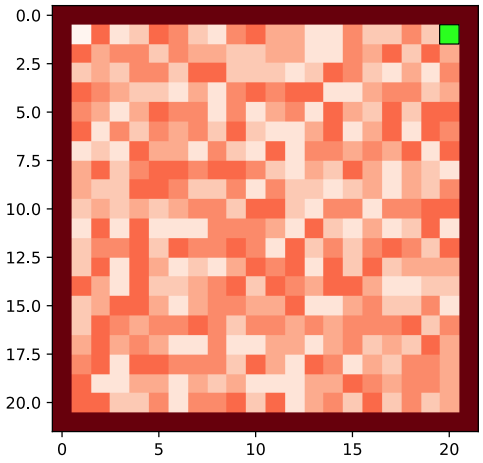

Figure 4.4: Map 4 — Random Pattern with horizontal Goal

The fourth map is the first map again, but this time the goal is shifted to $x = 20, y = 1$. With this, we want to observe how both approaches behave, when the most direct way to the goal would in theory be straight to the right, but this is prohibited by the maximum density criteria. This also checks if the algorithms can even find the goal if it is not at $x = 20, y = 20$. Here we can again expect that the map is at least solvable. Since we know that this would under normal circumstances be solvable by simply walking to the right, we can expect to see a high portion of moves in that direction mixed with the occasional north or south movement. With that said, the path, if it does not purely consist of moves to the right, can be no shorter than 21 moves. This number can be calculated by taking the x distance to the goal of 19 and adding at least two y-axis moves. The y-axis moves are the result of an object becoming too dense when only moving along the x-axis. This was tested manually on this map, and therefore we have to include two y-axis moves to move around the too dense obstacle.

5 Results and Evaluation

In this chapter, we will go over the results of both the EA and the A-Star algorithm. As a reminder, these results stem from 31 runs of the EA and one run of the A-Star algorithm on each map, respectively.

5.1 Map 1

The results for map 1 showed that each run the EA did, it reached the goal and finished with its best fitness score after between 1000 and 3000 generations. Here and in the following, we will look at the best run that the EA produced, since we are interested in a comparison between the best EA and the best A-Star path. Figure 5.1 illustrates the fitness progression per generation of the best run. We will only be looking at one graphic here because the other progression graphs are very similar. The graph displayed in figure 5.1 comes from the best run for map 1, which reached a final fitness score of 98.33 after 1011 generations. We can see that we have a strong increase in the fitness values between generations 0 and 200 and then only small bumps on a plateau. In this run, the goal was collected after 88 generations, so if we were only interested in a solution and not an optimal solution, we could stop the algorithm there. In figure 5.2, we can see the final path of this run. This path contains no repeating movements and no collisions, so it did not try to move into a wall or collide with objects that were too heavy to move. As for the length of the path, we have 38 steps, which is also the shortest path that we could get without any obstacles by alternating between going south and east. The bar chart in figure 5.3 illustrates the distribution of the final fitness values of the EA runs on map 1. We can

conclude that the best fitness value was only reached by one run. The box plot in figure 5.4 gives us a bit more insight into the values that were produced by the EA. We can see that the minimum is at 98.14 and the maximum is located at 98.33. The first quartile at 98.188 and the third quartile is at 98.273 which gives us an interquartile range of 0.085. Furthermore, the median of our values is at 98.225 while the mean is at 98.233. Now that we have seen the best result of the EA, we can take a look at what A-Star produced. In the figure 5.5, we can see the path that the A-Star algorithm chose for this map. We can see that this path is very close to the best path of the EA. They also achieved nearly the same fitness score, with the score of A-Star being 98.38. The path length is also 38 and with that again the best length without obstacles. The Fréchet Distance of these 2 paths is 2.236 Units and confirms that they are similar considering the map size of 20×20. As far as we can see, both algorithms were able to solve the problem in the given environment. Both approaches produced similar results, but A-Star was way faster and produced a more efficient path. If we look at Figure 5.3 we do not have a case of multimodality for the best fitness value that was reached. The best result for the EA was only reached once, and it also differs from the result for A-Star. This does not mean, that multimodality does not exist for this problem. The data we have gathered just does not contain any case of it. Showing that epistasis exists is an interesting task. As we explained in the chapter "Related Work", we would have a case of epistasis if the effect of one gene is suppressed by another gene. In an environment without movable obstacles, the effect of a gene could be considered as a change in the position of the agent. In our environment, this can be broadened to a change of the agents' position and a change in the map structure. Therefore, epistasis could only exist if 2 moves cancel each other out and make no structural change to the environment. On map 1, this would only be possible when the agent moves along the path that it has created and makes redundant moves, such as a step to the left and then back to the right on a horizontal part of the path. But this would mean that we have numerous redundant moves which increase the step counter and therefore decrease the fitness score of the individual, which leads to it no longer being propagated. All in all, even if epistasis would exist

at some point, the path containing it would be eliminated due to it having a worse fitness score. Figure 5.6 shows the final path of run 1 on the left and the final path of run 4 on the right. If we take a closer look at run 4, we can see that it made a redundant move near the start by going north and then south again. Other than that, both paths look at least kind of similar. The redundant move is where epistasis comes into play. It is not possible for the path on the left, to be morphed into the path on the right and be propagated. If we only look at the redundant move and imagine that through some mutation this move would be incorporated into the path on the right, this path simply would not be propagated since it would be at least one step longer and therefore worse than the path we already have. If we wanted to use the biological terms, we could say that the genome that would be created through this mutation would not be developed further since the already existing better genome would simply prohibit that, by being "fitter". This is the embodiment of survival of the fittest, and therefore the core principal the EA works on. With that said, we can make the general statement that epistasis exists, since a fitter path will always prohibit a less fit path from manifesting itself.

5.2 Map 2

The results of map 2 show that the best fitness values of all runs were reached between 1000 and 3500 generations and are thus similar to those of map 1. Figure 5.7 shows the fitness progression of run 18, which was the best run with a fitness score of 99.35. We have a strong increase in fitness value in the early generations and then a plateau with the last improvement in generation 2383. In this run, the goal was found for the first time in generation 35. Figure 5.8 shows the path of the best run. We can see that the EA found an almost perfect way through the valley, with one deviation where it went east instead of south. This mistake is corrected directly afterward by going south 2 times. Figure 5.9 contains the fitness distribution of the final EA fitness values on map 1. It shows that the highest fitness score was only reached once, and it

was much higher than the second-best score of 99.24. The box plot in Figure 5.10 gives us another view of the distribution of the fitness values. We can see that the minimum value is 98.81 and the maximum value 99.35. This can be confirmed by taking a look at the bar chart in Figure 5.9. The first quartile is located at 98.993 and the third quartile at 99.185. This gives us an interquartile range of 0.192. The median is at 99.085 and the mean at 99.084. As we have now looked at the results of the EA, we can examine the outcome that A-Star has produced. In Figure 5.11 we can see that the path A-Star chose for the valley is the perfect one. It has no energy requirement since it only moved through pockets of air and since the path goes directly from the start to the goal, it is the shortest path that can be taken here. With that, the path reaches a fitness score of 99.62 and therefore is better than the best EA result. The Fréchet distance of both paths is 1. As both outcomes have been examined, we can proceed to discuss multimodality and epistasis. Exactly as for the first map, we do not have evidence for multimodality. The best EA result was only reached once, and it is also not the best result over all, so we cannot say that multimodality is present for the best result. Regarding epistasis, we can make arguments similar to those of map one. It would be possible for the agent to make moves that cancel each other and have no impact on the environment. This could happen in two adjacent air pockets of the valley, for example. But even if it happened, the fitness score would decrease and the individual would be eliminated when a new population is generated.

5.3 Map 3

The runs for map 3 showed that the best EA results were achieved between generations 1000 and 3000. This wide margin can be explained by the random nature of the evolutionary algorithm. When we find a good path to the goal, the chance that we will improve it goes down for every further improvement, since there are less good paths that we can create through random mutations. The best fitness score was achieved by run 1 with a score of 98.88. Looking at the

path in figure 5.12, we see that it has an interesting resemblance to the paths of map 2. We shall revisit that matter during our discussion regarding the optimal route that the A-Star algorithm generated for this particular map. The first time the goal was reached in run 1 was in the 54th generation. As we can see in figure 5.13, the fitness progression of this run shares similarities with figure 5.1 and figure 5.7, which contain the figures for fitness progression for map 1 and map 2. A strong increase in the first 50 to 100 generations, with a following plateau and slight increases towards the end. Figure 5.14 shows the fitness distribution over all runs. Examining it, we can conclude, that the best fitness value of 98.88 was only reached once. Thus, there is no multimodality regarding the best result. This does not mean that there are no cases of multimodality for this result, but that we did not find any. By creating a box plot with our values (figure 5.15), we can see that the minimum value is at 98.55 and the maximum value at 98.88. Again, this can be confirmed by looking at Figure 5.14. Furthermore, we can see that the first quartile is at 98.633 and the third quartile at 98.765. Resulting from the difference between the third and first quartile, we get an interquartile range of 0.132. The mean is located at a value of 98.701 and the median at 98.695. The A-Star algorithm output in figure 5.16 shows that the best EA path and this one share plenty of similarities. The only difference we can see is on the last part of the path, where the EA chose to go south and A-Star stuck to a kind of zig-zag-pattern with a focus on going east. The Fréchet distance of the EA and the A-Star path is 1.41, showing that they are similar to each other, considering the map size of 20×20. With a fitness score of 98.91, A-Star performed better than the EA that reached a score of 98.88. As for the question of epistasis, the arguments are the same as the ones used for map 1 and 2. Even if it existed in one point of the evolutionary process, it would be eliminated going forward since the fitness score of the individual would be lowered.

5.4 Map 4

The final map that we intend to examine is the first map that has its goal shifted to the upper east corner, or in other words, horizontally towards the starting position. The best fitness score that was reached on this map was 99.08, and it was reached by 30 runs in total. This drastic change, when compared to the other maps, can be explained by the reduction of the distance to the goal. On all other maps, we had the goal positioned at the maximum distance to the start. Since the distance was so large, there were many possible variations that the path could follow. Here we shortened the distance to the goal significantly, and thus there are fewer variations that will improve the path itself. This leads to most of the runs taking the same path. One run that reached this fitness value is run 1. We can see this run in figure 5.17. We can see that the path consists of 1 step east, 1 south and then continued to the east until it went north and then east to the goal. This is what we predicted in the chapter "Experiments", being that the path will probably consist of two y-axis moves, with the remaining moves being x-axis moves. Figure 5.18 shows the fitness progression of the best EA run of map 4. We have a strong increase in fitness value in the first 70 to 80 generations, and then a plateau until we have the last improvement in generation 749. The best result was achieved between 350 and 1000 generations, which sets this map apart from the rest. This can be explained by the reduced distance between the start and the goal. Since we have less distance to traverse, we can reach the goal in less generations, since we need less mutations to get a goal path. Figure 5.19 shows the distribution of fitness values for the EA runs on map 4. We can see, that all but one run reached the best fitness value of 99.08. The one run that did not reach this value got a score of 99.07. This shows us, that we have a case of multimodality for the best reached value here, and we will examine this later. The box plot in figure 5.20 further illustrates the distribution of the fitness values. The minimum is located at 99.07 and the maximum at 98.08. Furthermore, the first quartile is located at 99.073 and the third quartile at 99.078. This gives us an interquartile range of 0.005. The median and the mean are both located at 99.075. Looking

at the A-Star path, this is the first instance in which both algorithms yielded the same outcome. Figure 5.21 shows the path that A-Star produced. This path achieves a fitness score of also 99.08 and a Fréchet distance of 0 to the result of the first EA run, confirming that both paths are the same. The fifth run of the EA has the same fitness score as run one of the EA and the A-Star run, but has a slightly different path. Figure 5.22 shows the first and the fifth run together. Even if it is only one different move at the start of the paths, we have a case of multimodality, since we have two different paths that share the same and also the best fitness score. The argument for epistasis is the same as for map 1, map2 and map 3.

5.5 Figures

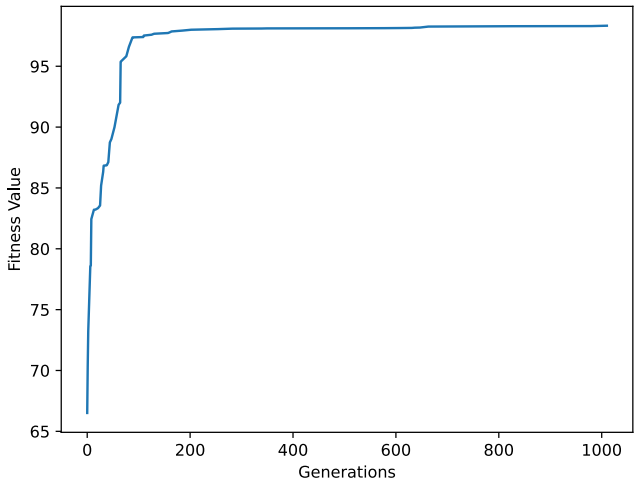

Figure 5.1: Map1 Run 1 Fitness Progression

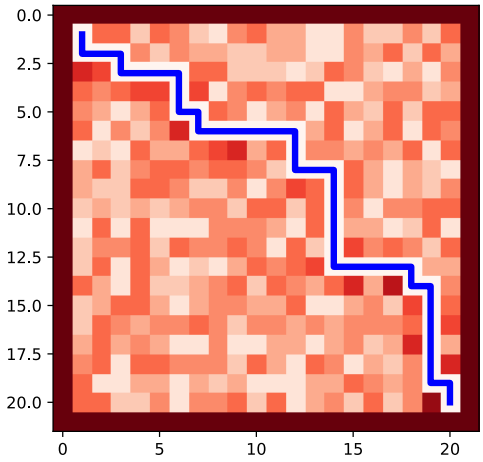

Figure 5.2: Map 1 Run 1 Final Path

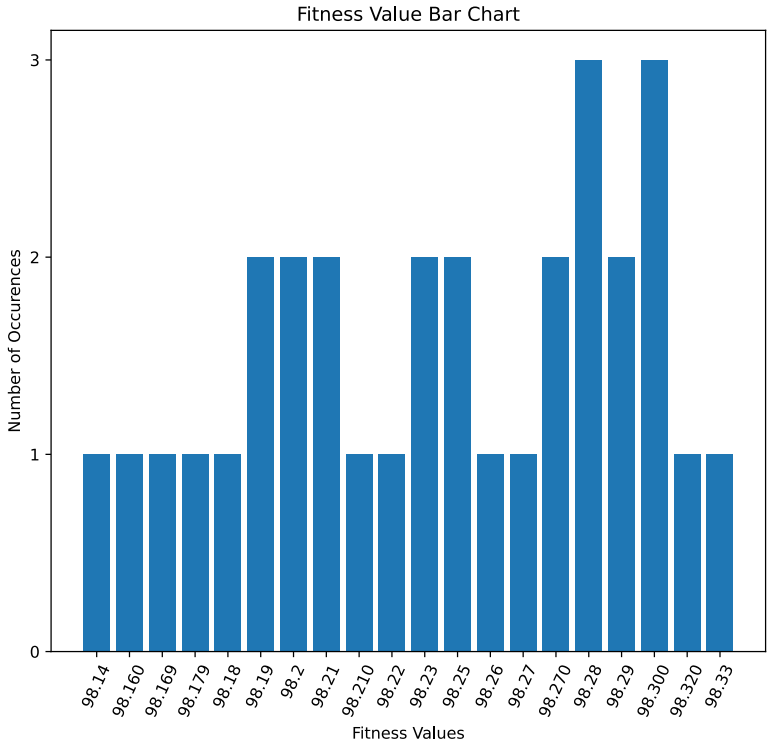

Figure 5.3: Map 1 Fitness Distribution

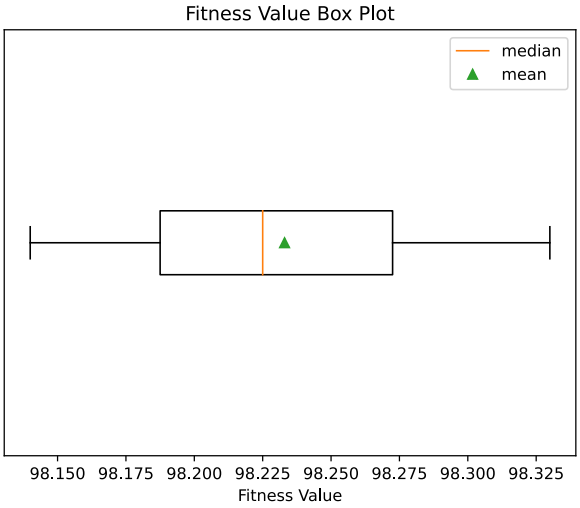

Figure 5.4: Map 1 Fitness Value Box Plot

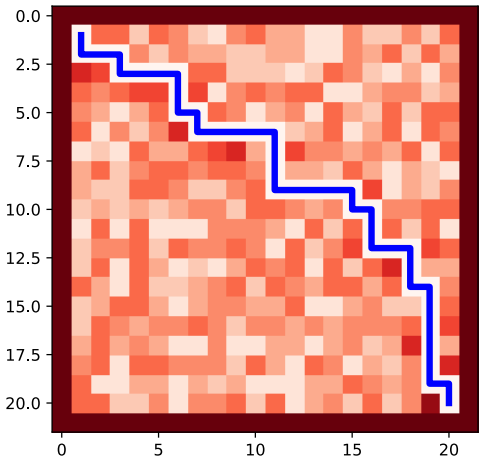

Figure 5.5: Map 1 A-Star Path

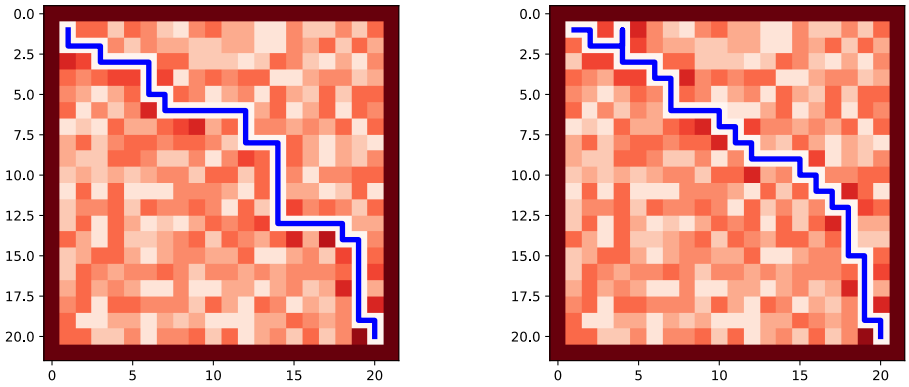

Figure 5.6: Map 1 Run 1 and Run 4

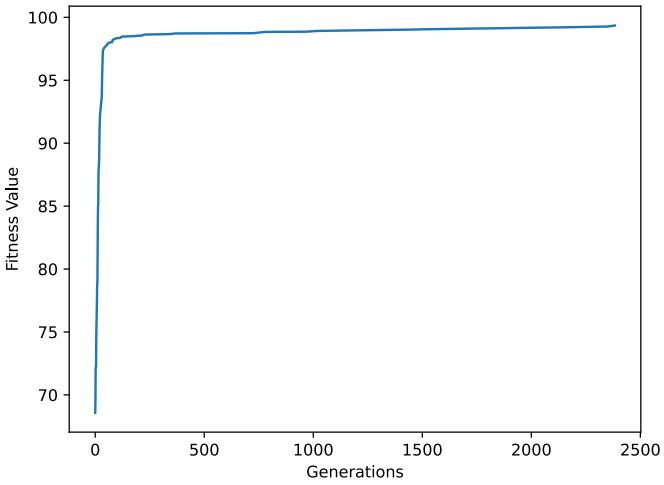

Figure 5.7: Map 2 Run 18 Fitness Progression

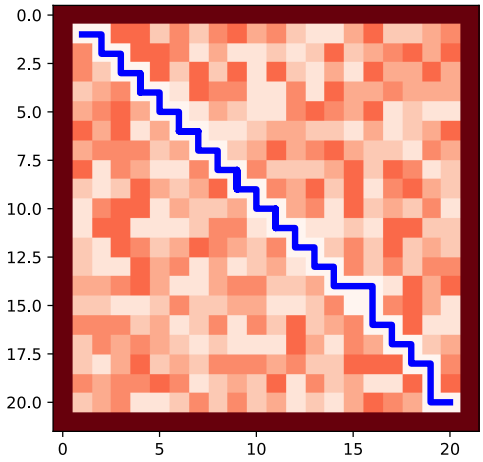

Figure 5.8: Map 2 Run 18 Final Path

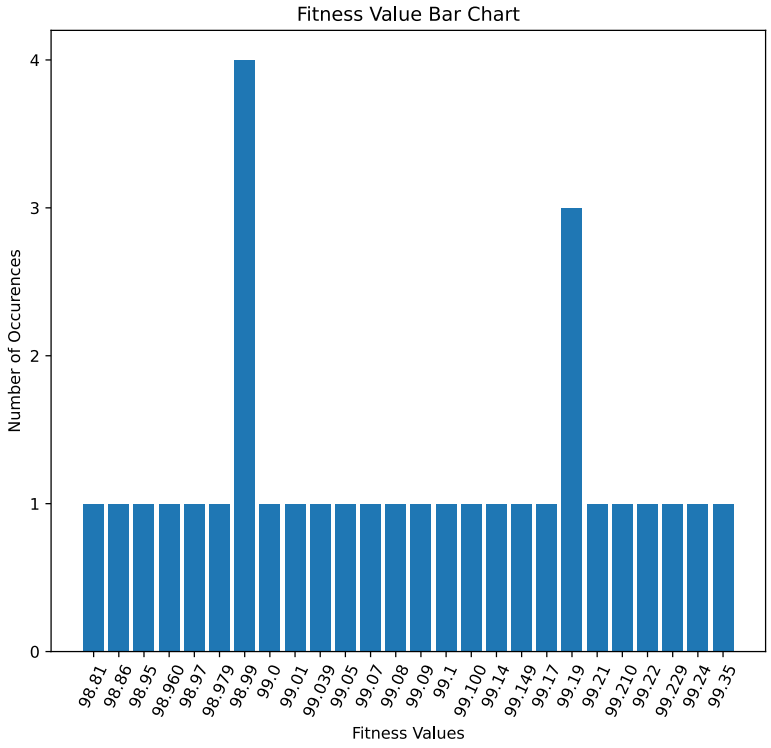

Figure 5.9: Map 2 Fitness Distribution

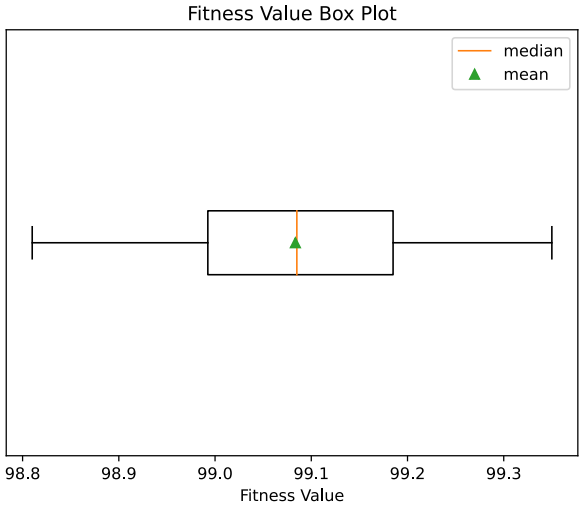

Figure 5.10: Map 2 Fitness Value Box Plot

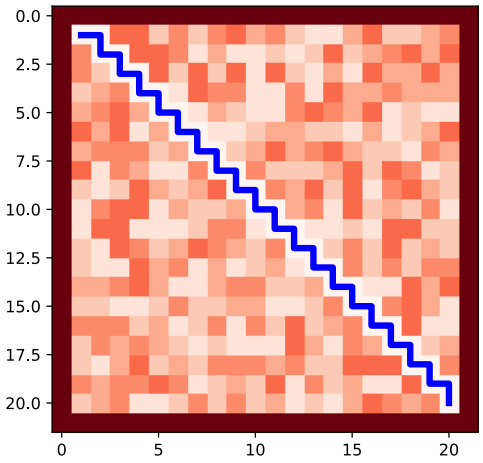

Figure 5.11: Map 2 A-Star Final Path

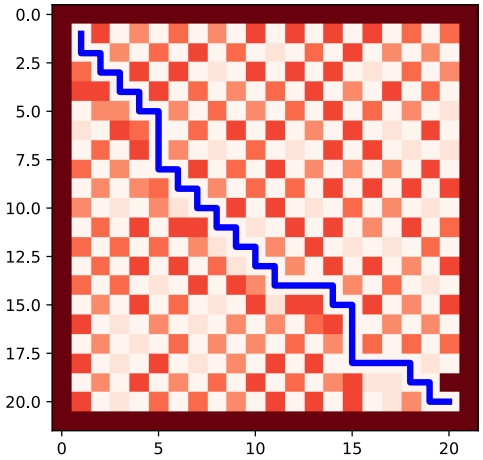

Figure 5.12: Map 3 Run 1 EA Final Path

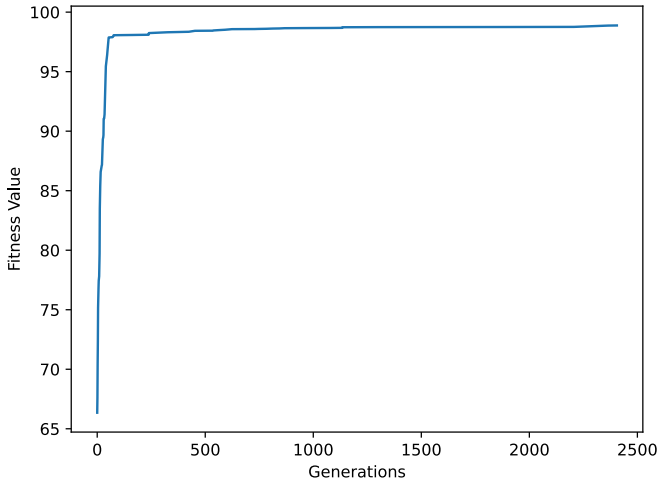

Figure 5.13: Map 3 Run 1 Fitness Progression

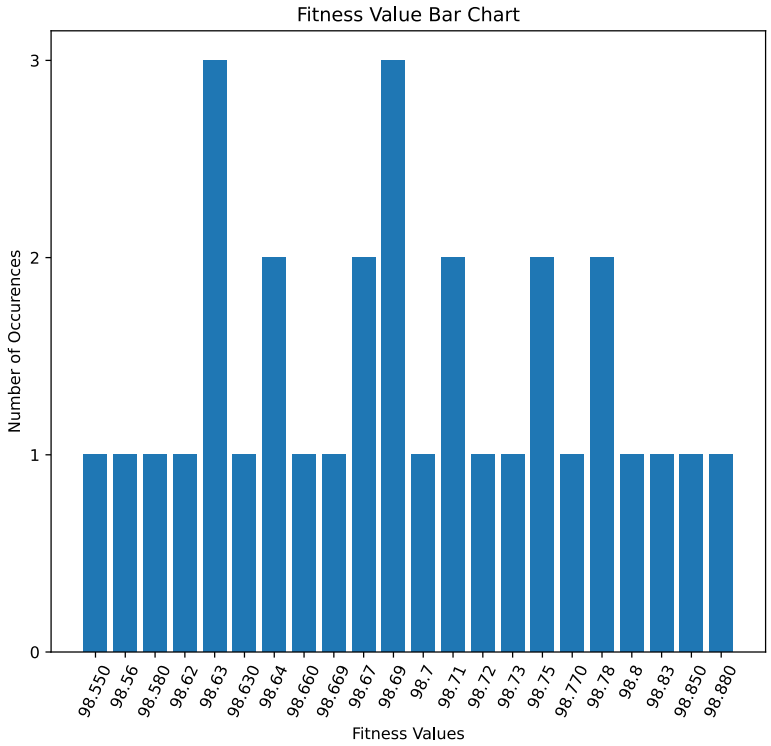

Figure 5.14: Map 3 Fitness Distribution

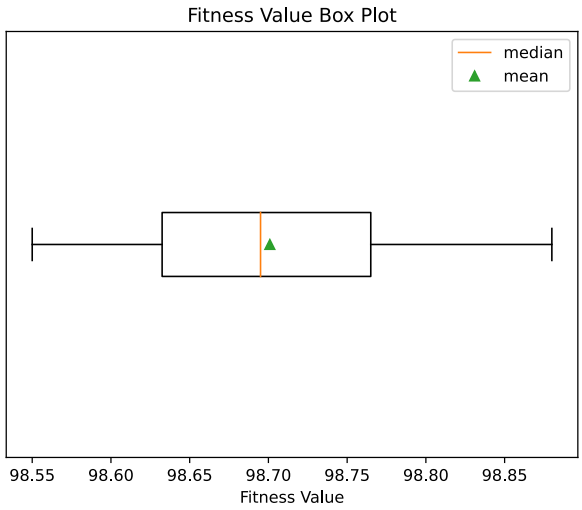

Figure 5.15: Map 3 Fitness Value Box Plot

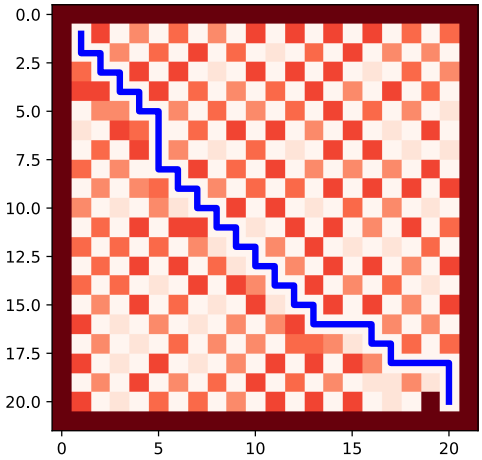

Figure 5.16: Map 3 A-Star Final Path

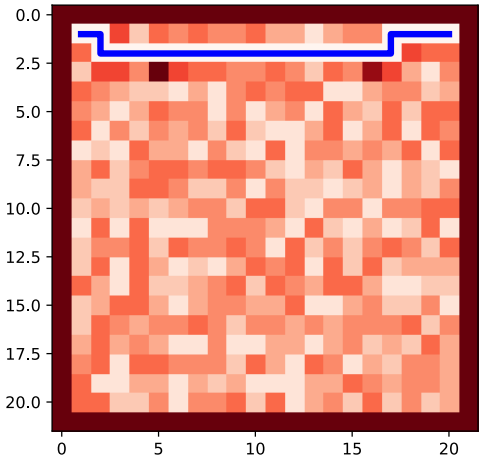

Figure 5.17: Map 4 Run 1 EA Final Path

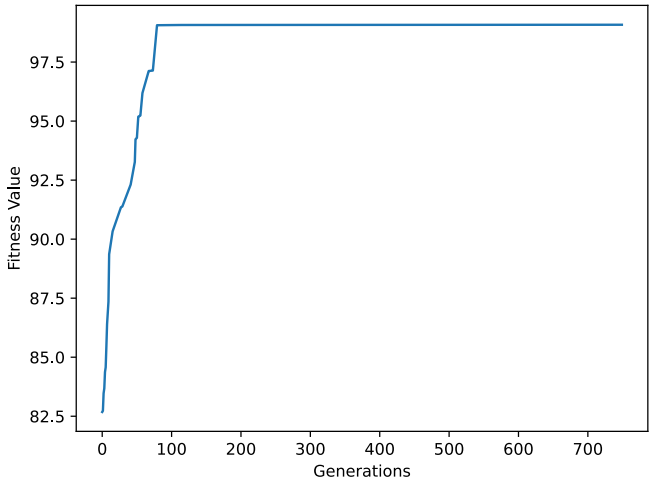

Figure 5.18: Map 4 Run 1 Fitness Progression

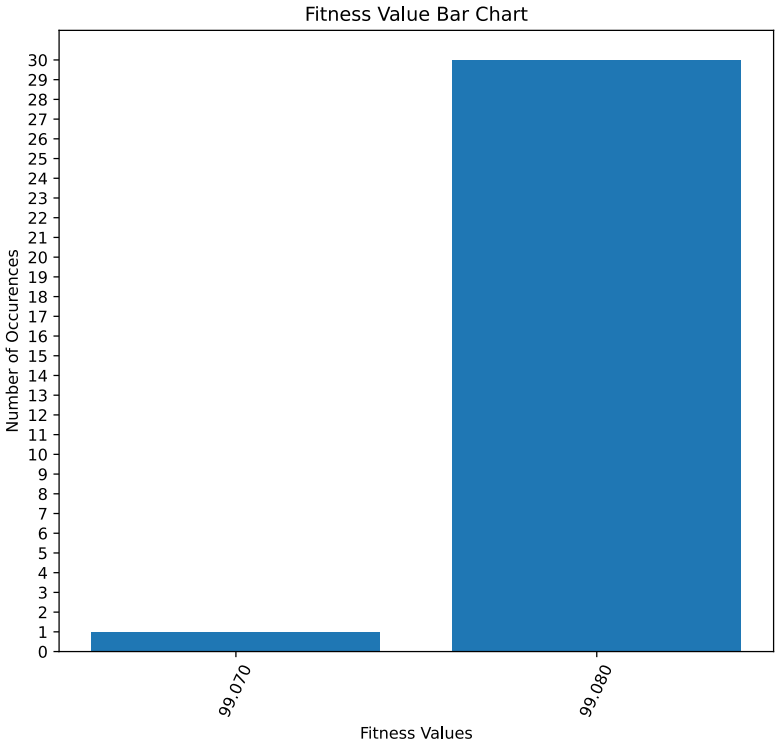

Figure 5.19: Map 4 Fitness Distribution

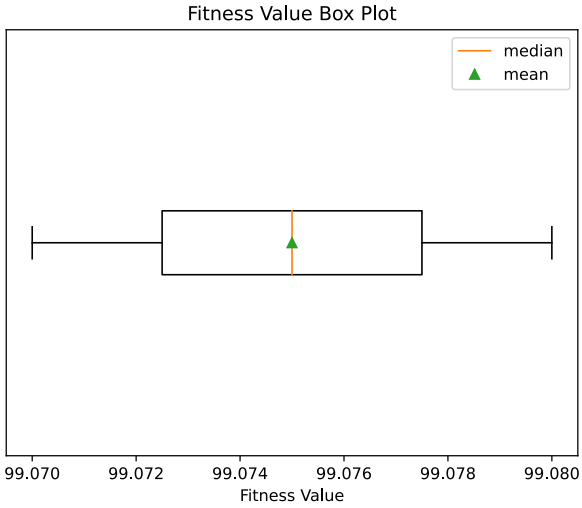

Figure 5.20: Map 4 Fitness Value Box Plot

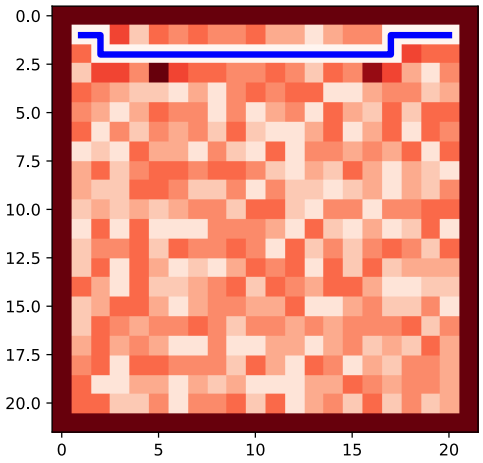

Figure 5.21: Map 4 A-Star Final Path

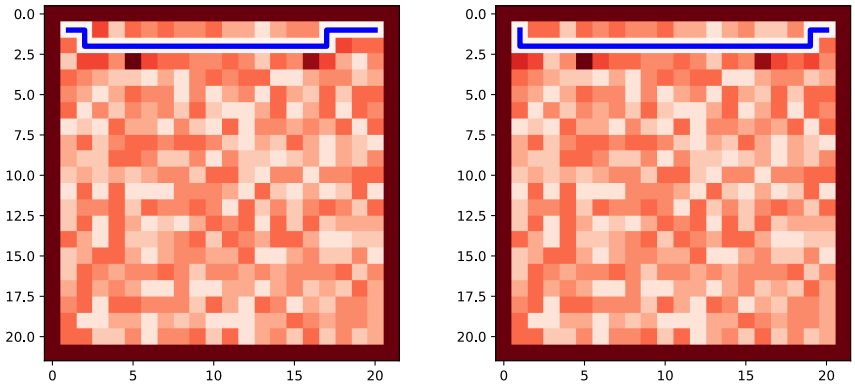

Figure 5.22: Map 4 Run 1 and Run 5

6 Conclusion

In this chapter, we will summarize the results of the evaluation from the previous chapter, and we will also discuss whether our research questions were answered. We will start with the most important question that we have: Are the approaches able to navigate a path influenced environment? The answer to that is yes. For all four maps that we tested, the evolutionary algorithm and the A-Star algorithm were able to reach the goal in every run that was performed. While this is the case, we also have to answer which of the two approaches performed better. In three of the four Maps, A-Star outperformed the EA, and the one map where they performed equally well was the one with the shortest path to the goal. Not only does it need just one run, but looking at the fitness values, it produced better results than the evolutionary algorithm. We have to look at how both algorithms work to answer the question of why that is the case. The evolutionary algorithm relies mostly on randomness. Any change in the path that happens, is the product of a mutation that is purely random in nature. This means that we will improve over time since we always select the best paths to be propagated, but if we only achieve minimally better results than the last generation, it will take more time to get to a good result. If we look at the A-Star algorithm, we see that it is the complete opposite. It does not rely on chance, but rather follows a methodical approach. This also allows us to run the A-Star algorithm only once, since we will always get the same result, while we have to run the EA several times and also let it iterate through many generations to produce the best result. This holds at least for bigger maps. With that said, we can come to the question of multimodality between paths. The respective best EA results on maps 1 to 3 were only reached once,

and the A-Star result was better than the best EA result each time, so we can conclude that we did not find a case of multimodality for the best EA results. We got the interesting results on map 4, which was the first map with the goal placed horizontally to the start. Here we had at least two runs that reached the best fitness value with different paths. This shows us that multimodality is possible, but we cannot say that it is possible on every map. This is the question of there being more than one best path. Especially in path influenced environments, this question is difficult to answer. As stated before, we have no reliable way to predict the best path or best paths for that matter. It is really likely that tinkering with the parameters of the fitness calculation also influences the number of perfect solutions that we can get. For example, giving higher penalties for path length and lower penalties for energy use might result in short paths with a high energy consumption. Since the only map on which we observed multimodality in the best result is map 4, and thus the map with the shortest path to the goal, we can assume that changing the penalties as described before, might lead to more cases of multimodality in for the best EA paths. With our setup, we can say that we definitely have multimodality at least with the conditions of map 4. The last question that we wanted to answer was that of epistasis. To simplify this question, without going too deep into biology, we could say that we ask if some paths cannot be developed when another path has already manifested. When looking at the results, the answer to that question became very apparent. Since we used an encoding, that directly responds to the path and used a setup for the EA that only propagated the best paths of each generation, we definitely have epistasis. With our approach, better paths will always prohibit worse paths from developing, since that is the principle our EA works on. Since we answered all the questions we proposed in the introduction to this thesis, we can move on to future work.

7 Future Work

In this chapter, we will discuss what research could be done on the topic of path influenced environments. One topic includes the question of how other established path finding algorithms perform on this problem, and what modifications would have to be applied to make them work. In this thesis, we used A-Star as an established pathfinding algorithm, but there are many more that can be tested in this kind of environment. It would also be interesting to know if there even is a method to approximate best paths in path influenced environments. A major problem for this thesis was that it was never clear if we found the best path, or if we just found one that was close. We, at least by now, cannot say that we found the best path for big maps with 100 percent certainty, so a method to verify this would be very helpful. What would also be interesting is the use of more than one agent moving through the map at a time. While the modification of the environment poses a challenge for even one agent, it would be interesting to see what would happen, if we have 2 entities that modify the map at the same time and possibly even interact with each other. Another topic would be if a map can possibly be "cleaned up". This means that we combine obstacles and move them out of the way until most of the map was cleaned and only some very heavy obstacles are remaining. With that in mind, we could pose the question of energy efficiency, so cleaning the map up while using a minimal amount of energy. Another basic, but interesting idea would be the use of non-square maps. Throughout this work, we only used square maps to test the different approaches, but it might be worth testing algorithms on maps that have small passages and other special structural features and evaluate how the different algorithms behave with that set of circumstances.

The last possible research topic that we want to discuss here is the possibility of "terraforming" the environment. Especially on map 1 and map 4, we started with an environment that was completely cluttered with obstacles. We moved through these maps once since our only concern was reaching the goal, but what would happen if we wanted to traverse it a few times. An example would be, that we collect the goal and then deliver it back to the start, or even do this a few times. For this, we would need an approach that clears a way for the entity to move through, so we do not need to move obstacles each time we want to traverse the same route.

Bibliography

[1] Nawaf Hazim Barnouti, Sinan Sameer Mahmood Al-Dabbagh, Mustafa Abdul Sahib Naser, et al. Pathfinding in strategy games and maze solving using a* search algorithm. *Journal of Computer and Communications*, 4(11):15, 2016.

[2] Thomas Bartz-Beielstein, Jürgen Branke, Jörn Mehnen, and Olaf Mersmann. Evolutionary algorithms. *WIREs Data Mining and Knowledge Discovery*, 4(3):178–195, 2014. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1124.

[3] William Bateson. Mendel's principles of heredity: Cambridge university press. *März 1909; 2nd Impr*, 3:1913, 1909.

[4] Ade Candra, Mohammad Andri Budiman, and Rahmat Irfan Pohan. Application of a-star algorithm on pathfinding game. *Journal of Physics: Conference Series*, 1898(1):012047, jun 2021.

[5] Heather J. Cordell. Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. *Human Molecular Genetics*, 11(20):2463–2468, 2002.

[6] Xiao Cui and Hao Shi. A*-based Pathfinding in Modern Computer Games. *International Journal of Computer Science and Network Security*, 11, November 2010.

[7] Kalyanmoy Deb. Multi-objective optimisation using evolutionary algorithms: an introduction. In *Multi-objective evolutionary optimisation for product design and manufacturing*, pages 3–34. Springer, 2011.

[8] M Maurice Fréchet. Sur quelques points du calcul fonctionnel. *Rendiconti del Circolo Matematico di Palermo (1884-1940)*, 22(1):1–72, 1906.

[9] MD Malkauthekar. Analysis of euclidean distance and manhattan distance measure in face recognition. In *Third International Conference on Computational Intelligence and Information Technology (CIIT 2013)*, pages 503–507. IET, 2013.

[10] Vo Ngoc Dieu Pandian Vasant, Gerhard-Wilhelm Weber. *Handbook of Research on Modern Optimization Algorithms and Applications in Engineering and Economics*. IGI Global, 2010.

[11] Bryan Stout. Smart moves: Intelligent pathfinding. *Game developer magazine*, 10:28–35, 1996.

[12] Jens Weise and Sanaz Mostaghim. Many-Objective Pathfinding Based on Fréchet Similarity Metric. In Hisao Ishibuchi, Qingfu Zhang, Ran Cheng, Ke Li, Hui Li, Handing Wang, and Aimin Zhou, editors, *Evolutionary Multi-Criterion Optimization*, Lecture Notes in Computer Science, pages 375–386, Cham, 2021. Springer International Publishing.

[13] Mohd Shahrizal Sunar Zeyad Abd Algfoor and Hoshang Kolivand. A comprehensive study on pathfinding techniques for robotics and video games. *International Journal of Computer Games Technology*, 2015:7–7, 2015.

Declaration of Authorship

I hereby declare that this thesis was created by me and me alone using only the stated sources and tools.

Malte F. Speidel Magdeburg, November 30, 2023