Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Complex Social Systems:
Modeling Agents, Learning, and Games

Project Report

Optimizing agent interaction in pedestrian model ...

Joël Andenmatten & Jan Grunder & Damian Moser

Zurich
December 2021

Agreement for free-download

We hereby agree to make our source code for this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Joël Andenmatten Jan Grunder Damian Moser

Contents

1 Abstract

In this paper we used genetic algorithm to determine parameters for the social forces model. This was done using two fitness functions, one depending on how close and fast agents got to their destination and the other depending on how many collision occurred. The training leads to the result that in some scene, ignoring all interactions with other pedestrians and obstacles gives the best performance. Those parameters do not generalize well to scenes which they weren't trained on. This leads to the conclusion that in the combination of social forces model with genetic algorithm the fitness function needs to be independent of reaching high throughput. This finding stands in conflict with our original goal to find an efficient evacuation algorithm using the proposed models.

2 Individual contributions

Every person in the group was involved in coding and writing the project report.

The general model was implemented by Jan and Damian with changing it to numpy being done by Joël. The field of view extension was written by Jan. The UI was written by Joël (Plotting) and Jan (Playback). The serialization of our simulation was done by Damian. Writing the genetic algorithm, choosing its size and fitness function. The code for using the distributed abilities of the framework was written by Jan.

The Introduction, Motivation and the section on Related work was written by Jan. The model was described by Joël. The implementation was written by Damian. The result section was written by both Damian and Jan. The summary was written by Damian and the outlook section was written by Jan.

3 Introduction and Motivations

Nowadays the field of robotics is bigger than ever before. Robots, whose main use case was in highly specialized manufacturing, are becoming more mobile and are also starting to become a part of everyday life. Be it as a simple cleaning robot, a humanoid or a self driving car. Being a product of newest technology they also are expensive. Due to that, it is an interesting venue to not only evacuate humans but also the robots in case of an emergency.

In the field of robotics the general question of navigation has been one of the biggest researched topics. It used to be that direct interaction and sharing of space with humans was not considered as important because robots were not as common and were used in controlled environments with strict safety rules in place for people to follow. This has changed as robots are becoming more commonplace and therefore the interest in human-aware navigation has grown. This is the reason why we choose the social forces model, as then the machines consider people and keep a certain distance from them, as well as moving in a way humans can predict, which both could lead to better human robot interactions and safety. Since we not only want to mimic humans, we gave the robots the goal to be as efficient as possible while still avoiding collisions

Other work [2, 3] has implemented the social forces model for very specific robots with fixed parameters. We are using a genetic algorithm so that the model could be tuned for arbitrary robots.

4 Related Work

The paper concerned with robot companions [2] proposes a novel navigation and interaction scheme for services robots which are following a target. It is based on the social forces model with parameters and weights for the different forces. The weights were evaluated experimentally (not simulated) by getting direct user feedback. The robot takes the predicted trajectories of other pedestrians into account, also by using the social forces model and probabilistic destination prediction.

Another paper [3] expands the social forces model, already used for walking/driving robots, to the third dimension for uses in drones. The goal is also following a human. Same as [2] this model also uses "proxemics" defined in [4] as quantitative metrics to evaluate the performance of their robot. This is done by defining different areas in which robot should and should not be and rewarding and punishing appropriately.

5 Description of the Model

5.1 Social Force Model

For the base of the model we use the social force model as described by D. Helbing in [9]. We create agents with different properties such as locations, destinations and velocities. Each agent heads to the direction of its destination but it also will

experience forces coming from agents and objects in its surroundings. Therefore we note the social force of a pedestrian α as:

$$\vec{f}_\alpha(t) = \frac{1}{\tau_\alpha}(\vec{v}_{\text{desired}} - \vec{v}_\alpha) + \sum_{\beta \neq \alpha} \vec{f}_{\alpha\beta}(t) + \sum_i \vec{f}_{\alpha i}(t), \tag{1}$$

where $\vec{f}_{\alpha\beta}$ is the interaction force between two pedestrians and $\vec{f}_{\alpha i}$ the interaction force between an obstacle and the pedestrian. With τ_α we note the relaxation time, with $\vec{v}_{\text{desired}}$ the velocity in desired direction that the pedestrian tries to reach and $\vec{v}_\alpha$ is the current speed of the pedestrian. Our model uses robots for pedestrians and we expect $|\vec{v}_{\text{desired}}|$ to be the same value for all robots.

We need the pedestrians to care about collision especially while making a step, since the stretched out leg deforms the circular appearance of a pedestrian to an elliptic shape and makes collision more likely. To accommodate this behaviour we use the elliptical specification. For the following equations we define $\vec{d}_{\alpha\beta} = \vec{r}_\alpha - \vec{r}_\beta$, where $\vec{r}$ is the position vector of a pedestrian α respectively β. First we simplify the interaction force and convert it to the gradient of a potential

$$\vec{f}_{\alpha\beta}(t) = \vec{f}(\vec{d}_{\alpha\beta}(t)) = -\nabla_{\vec{d}_{\alpha\beta}} V_{\alpha\beta}(b_{\alpha\beta}), \tag{2}$$

which is based on a exponentially decaying function

$$V_{\alpha\beta}(b_{\alpha\beta}) = AB \exp\left(\frac{-b_\alpha \beta}{B}\right), \tag{3}$$

where A is the interaction strength and B the interaction range. The variable $b_{\alpha\beta}$ is the semi-minor axis of the elliptical equipotential lines, which is specified by

$$2b_{\alpha\beta} = \sqrt{(||\vec{d}_{\alpha\beta}|| + ||\vec{d}_{\alpha\beta} - (\vec{v}_\beta - \vec{v}_\alpha)\Delta t||)^2 - ||(\vec{v}_\beta - \vec{v}_\alpha)\Delta t||^2}, \tag{4}$$

and combines to a pedestrian interaction force of

$$f_{\alpha\beta}(\vec{d}_{\alpha\beta}) = A \exp\left(\frac{-b_{\alpha\beta}}{B}\right) \cdot \frac{||\vec{d}_{\alpha\beta}|| + ||\vec{d}_{\alpha\beta} - \vec{y}_{\alpha\beta}||}{2b_{\alpha\beta}} \cdot \frac{1}{2}\left(\frac{\vec{d}_{\alpha\beta}}{||\vec{d}_{\alpha\beta}||} + \frac{\vec{d}_{\alpha\beta} - \vec{y}_{\alpha\beta}}{||\vec{d}_{\alpha\beta} - \vec{y}_{\alpha\beta}||}\right), \tag{5}$$

where $\vec{y}_{\alpha\beta} = (\vec{v}_\beta - \vec{v}_\alpha)\Delta t$.

The interaction force between obstacles and pedestrians $\vec{f}_{\alpha i}$ is derived similar to $\vec{f}_{\alpha\beta}$ in Equations 2 and 3, with the difference that we use the circular instead of the elliptic specification and replace the semi-minor axis $b_{\alpha\beta}$ with the shortest distance

between pedestrian and obstacle $||\vec{d}_{\alpha i}||$. This results in the interaction force between obstacles and pedestrians

$$f_{\alpha i}(\vec{d}_{\alpha i}) = C \exp\left(\frac{-||\vec{d}_{\alpha i}||}{D}\right) \cdot \frac{||\vec{d}_{\alpha i}|| + ||\vec{d}_{\alpha i} + v_\alpha \Delta t||}{2||\vec{d}_{\alpha i}||} \cdot \frac{1}{2}\left(\frac{\vec{d}_{\alpha i}}{||\vec{d}_{\alpha i}||} + \frac{\vec{d}_{\alpha i} + \vec{v}_\alpha \Delta t}{||\vec{d}_{\alpha i} + \vec{v}_\alpha \Delta t||}\right), \tag{6}$$

where C is the interaction strength and D the interaction range.

5.2 Genetic Algorithm

Genetic Algorithms are a type of Evolutionary Algorithms and are used to solve optimization problems. They are counted to areas of computational intelligence and artificial intelligence. The inspiration for this algorithm was drawn from the work *On the Origin of Species* by Charles Darwin. The idea is to treat parameters, that need to be optimized, as attributes/DNA of an individual.

To begin, a population of these individuals is created and all individuals have random parameters. This is treated as the first generation. First of all the individuals are evaluated for their *Fitness*. We then select a parental population which includes some of the fittest individuals and also some chosen by randomness. That generation then goes through *Crossover* and *Mutation* to form the next generation. *Crossover* is combining the parameters of two or more individuals to create a new individual and *Mutation* is often implemented using a probability function that changes a attribute of an individual to a random value. In that way the Genetic Algorithm is optimizing the fitness of individuals generation after generation.

This process of simulating generation after generation either terminates after a predefined number of iterations or by the convergence of the fitness function. [7]

6 Implementation

You can find all the code and generated files at github.com/madebydamo/css. Our model is written in python using github.com/numpy/numpy as linear algebra library. The optimization is done by github.com/DEAP/deap, a evolutionary computation framework.

6.1 Manual Parameters

In the process of creating a simulation with good results, we had to tweak a few boundary conditions by our self, which we discuss in this section.

6.1.1 Simulation

A simulation of a scene is 10 seconds long with 30 frames per second. Therefore we update our agents 300 times per simulation. Our scenes are designed so that they can be completed within 10 seconds. They are only 10 by 10 meters and contain at most 12 agents and a few objects. A agent by itself has a radius of 0.25 meters. This value is used to calculate collisions between agents or agents and walls.

6.1.2 Genetic algorithm

We trained with generations of size 100. Since we saw in the testing phase, that it already settles on parameters after e few generations, we only calculated 50 generations in our result sets. Everything above did not yield us better results. We tried to improve variety of later generations by setting the random mutations of parameters to 10%. Since DEAP has multi-core processing implemented by default, we could run the simulations in a reasonable amount of time on our local machines.

6.1.3 Heuristic for genetic algorithm

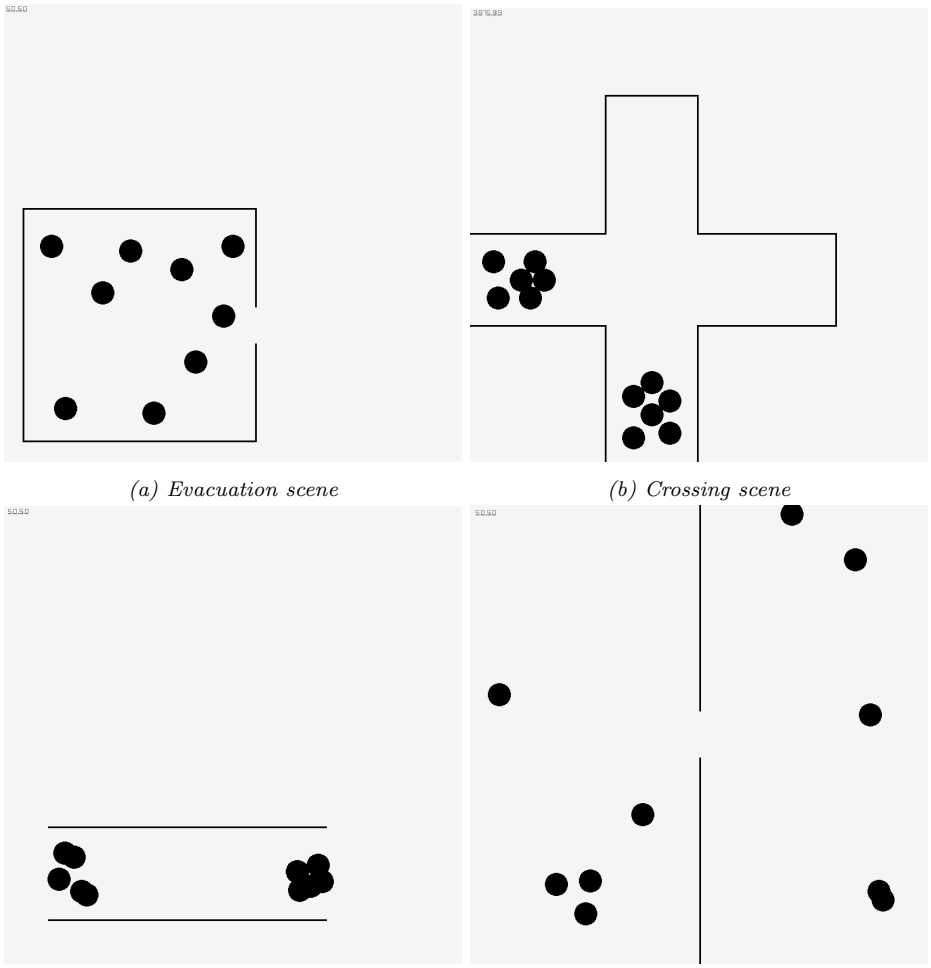
To determine between good and bad agents, we had to give DEAP a fitness function. While simulating a scene, we determine , how close to their destination the agents get.If the agents reach their destination, they are additionally rewarded for how fast they were. Otherwise they only receive points for how close to the destination they managed to get. To punish collisions, we also return a second value for the number of collisions. We configured DEAP such that it tries to maximize the first fitness value and minimizes the second.

6.2 Algorithmically determined parameters

The parameters that we optimized using the genetic algorithm are the interaction strength A and interaction range B of the interaction force between agents. Also we have interaction strength C and interaction range D of the interaction force between agents and objects. Finally we have parameter τ, which determines how fast we change our velocity.

6.3 Scenes

To train our agents in an environment we needed some scenes shown in Figure 1. Crossing and Evacuate where used to train our model. Bottleneck and Lane were used to test our generated data.



(a) Evacuation scene

(b) Crossing scene

(c) Lane scene

(d) Bottleneck scene

Figure 1: Different scenes visualized using ui.py

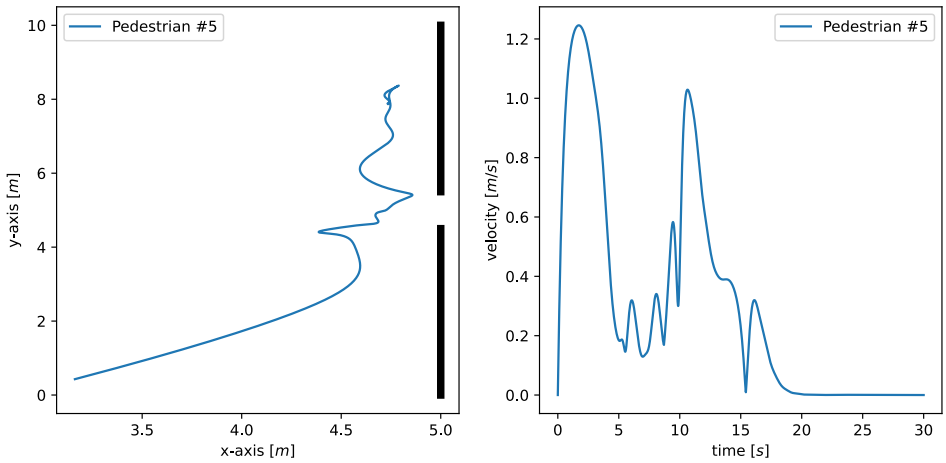


Figure 2: Pedestrian stuck infront of a wall

7 Simulation Results and Discussion

All the data we collected using our model is presented and discussed in this section.

First we ran the scene "bottleneck" using the parameters of D. Helbing [5] to verify that our implementation of the model works. We also wanted to test whether or not we observe pedestrians blocking each other at a narrow door. This can be seen in Table 1 with more pedestrians from group A arriving at their target destination than pedestrians of group B. Interestingly, some pedestrian do not reach their goal at all, ending up somewhere close to the wall as seen in Figure 2. They reach the correct height but cannot get around the obstacle. This could be solved as seen in [6] by adding an additional force which guides pedestrians around walls and obstacles.

Time until destiantion	< 5 sec.	< 10 sec.	> 10 sec.	does not reach
Group A	1	5	2	2
Group B	1	2	3	4

Table 1: Simulation on 20 pedestrians

We had some troubles with our agents minimizing collisions. While collision minimization would be critical for a real world use case with robots, we probably have not set the weight of collision minimization correct in our genetic library. Even though the amount of collisions shrinks while training, it never reaches our desired zero. In Figure 3 we can see that it accepts more collisions as a trade of for a better

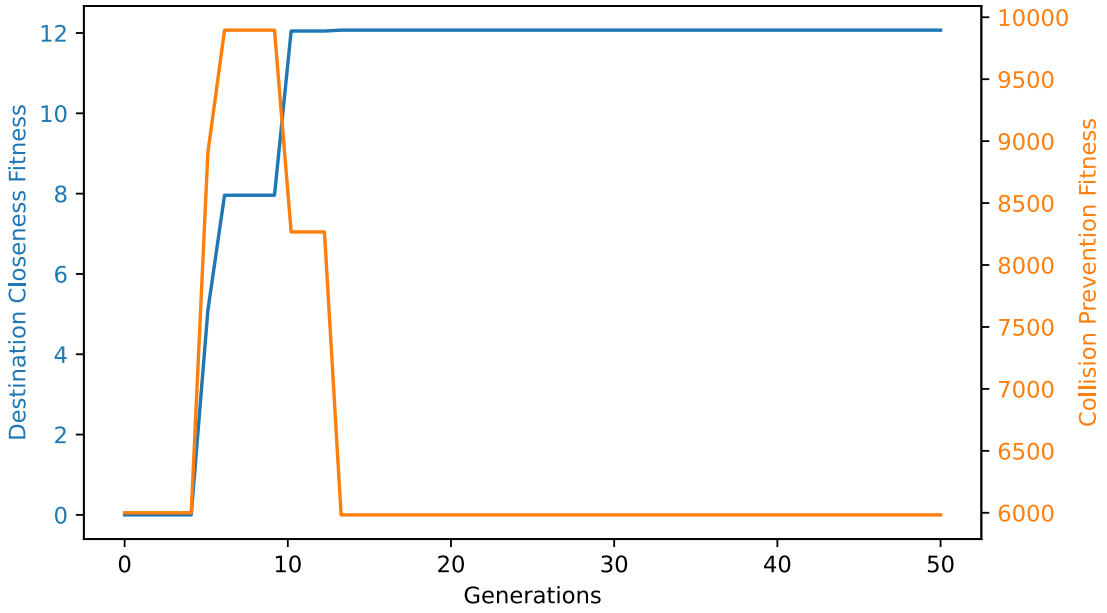


Figure 3: Fitness Values of generation 50 of crossing simulation.

destination closeness fitness. It could be that the overall fitness decreases from one generation to another. Furthermore it looks like we are stuck in a local maximum. The parameters themselves do not change as frequently as in earlier generations as we can see in Figure 4. It is also notable, that the only parameters that changes after generation 15, are the ones who are not changing the outcome of the simulation. For example if the interaction strength A is set to zero, the interaction range B does not matter anyway and vice versa. It is also important to note that we received similar results for several training instances with forces being neglected.

We trained two sets of parameters Evac and Cross. Set Evac was trained on the evacuation scene and Set Cross was trained on the crossing scene. We did not use the Lane and Bottleneck scenes for training in order to have some scenes where none of the parameters have been tuned. Table 2 lists the parameters for the simulated scenes next to the reference values from D. Helbing [5] and M. Capelle [1].

In Table 3 we see the resulting fitness values for different sets of parameters.

Due to the higher strength of the repulsive effect between agents in Capelle compared to Helbing, the agent simulated with those parameters are often further away from the goal after our time interval of 10 seconds. Interestingly it does not nec-

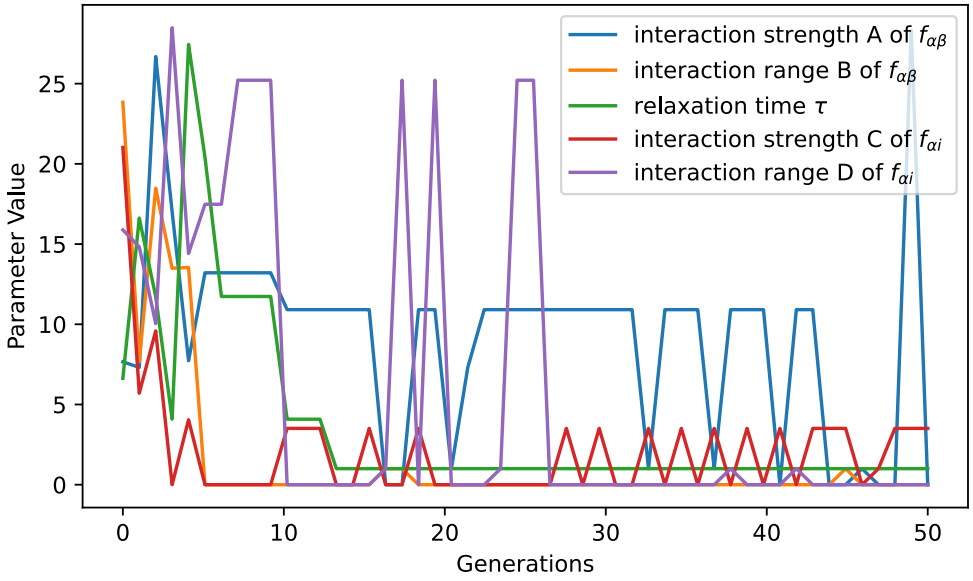


Figure 4: Parameter values of best agents in crossing simulation.

Source of Parameters	Evac	Cross	Helbing	Capelle
interaction strength A	0	0	2.1	21
interaction range B	0	0	0.3	0.3
interaction strength C	1	3.5	10	10
interaction range D	1	0	0.2	0.2
relaxation time τ	0.13	1	0.5	0.5

Table 2: The parameters that resulted with the generic algorithm in comparison to reference values.

Scene	Evacuate	Crossing	Lane	Bottleneck
Training set Evac	**9.09**, **1632**	**12.08**, 5603	**10.09**, 4016	**10.06**, **2646**
Training set Cross	9.07, 1941	12.07, 5983	10.08, 4939	10.05, 3146
Helbing	8.84, 2007	12.07, **2577**	8.77, 3201	8.77, 3201
Capelle	6.84, 2253	4.69, 4062	7.38, **3034**	3.46, 3012

Table 3: Fitness comparison using different sets of parameters and scenes. The first value (destination closeness) needs to be maximized and the second value (collision prevention) needs to be minimized. The best value in each scene is ***bold****.*

essarily decrease the amount of collisions. Both our trained sets completely ignore the repulsive force between agents. This explains the higher values in destination closeness fitness due to them just taking the direct path to their destination. Even though the set Cross ignores the forces between agents and walls, this does not give them any noticeable benefit due to the scenes not having paths that face directly into a wall. Cross also has a higher relaxation time τ and thus agents take longer to reach their desired velocity.

Interestingly ignoring the forces between agents seems beneficial to the collision prevention fitness in some scenes, as the Evac set has the lowest fitness in the scenes Evacuate and Bottleneck. A possible explanation is that agents stopping leads to collisions with other agents, not able to respond quickly enough. The reason Evac outperforms Cross is not clear to us.

In general Evac, which was trained on the scene Evacuate, does not generalize. It is outperformed by Helbing's parameter (which were chosen to correspond to human behavior). Interestingly it also performs well in the scene Bottleneck which could be due to this scene and Evacuation being similar, with the difference that in the bottleneck scene we have pedestrians going into both directions.

The parameter set Cross did perform worse than Helbing, which shows that Cross did get stuck in a local minima i.e. maxima during training. The fitness in scene Lane and Bottleneck are to be considered with caution due to them being produced procedurally. Training on those scenes results in irregular changes to fitness and it does not converge as seen in Figure 5.

8 Summary and Outlook

8.1 Summary

Our most notable result is that social and object forces are reduced to really small values and therefore not significantly change the behaviour of our agents based on its surrounding. This way they do not get slowed down by such forces and therefor optimize the speed of reaching their final destination. It seems that our agents managed to exploit the way the two fitness functions are combined. On one side we probably have not tuned our fitness function enough, on the other side it makes sense that the social forces model is not well suited for reaching both high speeds and minimizing collisions. It is inspired by human to human interaction, which has a social background and does not care about speed. And therefore it is not a surprise

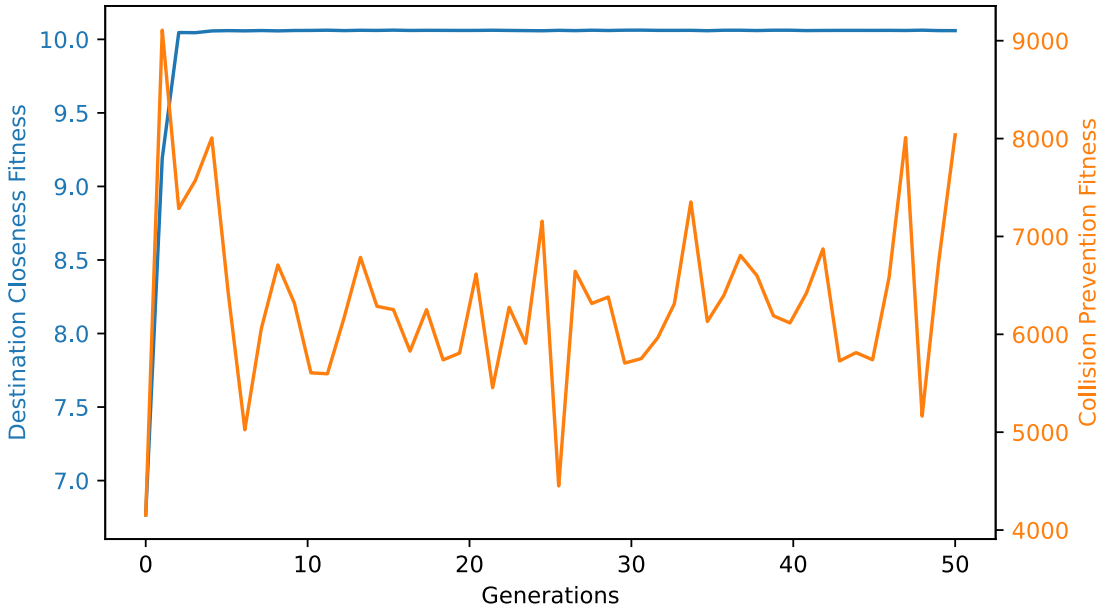


Figure 5: Fitness Values of generation 50 of lane simulation.

that speed optimization dismantle more or less all social interactions who slows them down.

8.2 Outlook

While our model performed rather poorly, one might improve the fitness function or try to optimize for other goals instead of speed. Maybe a more social goal like preserving a reasonable distance between agents might work better [4]. Even with the genetic algorithm running on multiple processors, the simulation in and of itself is quite slow. When incrementing the number of agents in a scene, we get a noticeable slowdown. This should be concern for future work, as not to be a limiting factor for its possible implementation on real hardware. An improvement in speed could be achieved making use of vectorized instructions using numpy. A parallel implementation of the social forces model [8] or mixing the social forces model with cellular automata [6] could not help speed up training but the running of the simulation on actual hardware.

The genetic algorithm has the potential for more parameters to be introduced. Other papers [2, 3] often weight the different forces using additional parameters.

Adding a parameter for the field of view of the robot could give insight into how much the robot has to see and if 360 degree vision, as used throughout this paper, is truly necessary.

9 References

[1] Max Capelle. "Validation and implementation of the social force model for crowd behavior". In: (2018).

[2] Gonzalo Ferrer, Anaís Garrell, and Alberto Sanfeliu. "Robot companion: A social-force based approach with human awareness-navigation in crowded environments". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013, pp. 1688–1694. DOI: 10.1109/IROS.2013.6696576.

[3] Luis Alberto Garza Elizondo. "Social-aware drone navigation using social force model". PhD thesis. UPC, Facultat d'Informàtica de Barcelona, Departament de Ciències de la Computació, Oct. 2016. URL: http://hdl.handle.net/2117/104437.

[4] E.T. Hall. *The Hidden Dimension*. A Doubleday anchor book, A609. Doubleday., 1969. ISBN: 9780385084765. URL: https://books.google.ch/books?id=zGYPwLj2dCoC.

[5] Dirk Helbing and Peter Molnar. "Social force model for pedestrian dynamics". In: May 1995.

[6] Anna Kormanová. "Combining Social forces and Cellular automata models in pedestrians' movement simulation". In: (2012).

[7] Oliver. Kramer. *Genetic Algorithm Essentials*. 1st ed. 2017. Studies in Computational Intelligence, 679. Cham: Springer International Publishing, 2017. Chap. 1-2. ISBN: 3-319-52156-X.

[8] Michael Quinn and Ronald Metoyer. "Parallel implementation of the social forces model". In: *Proceedings of the Second International Conference in Pedestrian and Evacuation Dynamics* (Jan. 2003).

[9] *Social Self-Organization : Agent-Based Simulations and Experiments to Study Emergent Social Behavior*. Berlin: Springer, 2012. Chap. 3. ISBN: 9783642240041.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Optimising agent interaction in pedestrian model

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**
Andenmatten	Joël
Grunder	Jan
Moser	Damian

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**
Zurich, 5. Dec 2021	J. Andenmaten J. Grunder Moser

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.