




# Evolving Artificial Neural Networks for Simulating Fish Social Interactions

Lea Musiolek<sup>1,5</sup>(✉) , David Bierbach<sup>2,5</sup> , Nils Weimar<sup>3</sup>, Myriam Hamon<sup>4</sup>,  
Jens Krause<sup>2,5</sup> , and Verena V. Hafner<sup>1,5</sup> 

<sup>1</sup> Adaptive Systems Group, Humboldt-Universität zu Berlin, Berlin, Germany  
[lea.musiolek@hu-berlin.de](mailto:lea.musiolek@hu-berlin.de)

<sup>2</sup> Department of the Biology and Ecology of Fish, Humboldt-Universität zu Berlin,  
Berlin, Germany

<sup>3</sup> Zoology Department, Universität Bonn, Bonn, Germany

<sup>4</sup> Bernstein Center for Computational Neuroscience, Berlin, Germany

<sup>5</sup> Science of Intelligence, Research Cluster of Excellence, Marchstr. 23, 10587 Berlin,  
Germany

<https://www.scienceofintelligence.de>

**Abstract.** Can we use computational modeling to infer whether fish can remember or anticipate each other’s movements? What minimum of temporal input and internal complexity is sufficient to model a specific fish, or to produce generally “fish-like” behavior? Agent-based modeling to emulate biological behavior has been used to great effect, both in real-world and simulated experiments. We present feedforward neural network architectures for simulating fish social interactions, evolved using evolution strategies in two different experiments. Evolution of the temporal input of the partner fish’s position when testing models on labeled data uncovers anticipation or memory capacities used by a focal fish. When testing via a general discriminator for fish-like trajectories, the right neural network architecture and temporal input are shown to be a necessary, but insufficient condition for highly lifelike simulations. Lifelike simulations for some datasets are possible as simple functions of the input, showing variability in the complexity of individual fish’s social behaviors.

**Keywords:** Evolution Strategy · Fish · Social Interactions ·  
Agent-Based Modeling · Artificial Agents

## 1 Introduction

Can a small freshwater fish anticipate a partner’s movements in social interaction? Does it need to, in order to behave “like a fish”? Finding the answers is

Funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany’s Excellence Strategy - EXC 2002/1 “Science of Intelligence” - project number 390523135.

**Supplementary Information** The online version contains supplementary material available at [https://doi.org/10.1007/978-3-031-56852-7\\_10](https://doi.org/10.1007/978-3-031-56852-7_10).

not as easy as it sounds. It has been shown through tailored biological experiments that *Poecilia reticulata* (guppies) are capable of leading and following each other, socially learning food locations, anti-predator behavior and other useful skills [7], choosing interaction and mating partners based on past observations of conspecifics [3], and learning another's movement patterns in order to precede it to a goal destination [2]. However, inferring from such behaviors that the fish are socially anticipating requires making many assumptions about their motivations and information processing. Pezzulo [17] and others define anticipation as using "predictive capabilities to optimize behavior and learning to the best of [one's] knowledge". In this study we show how computational modeling and evolution strategies can be leveraged to find out how guppies use temporal information about a partner to inform their actions *a posteriori* from recordings of freely moving fish pairs, without experimental manipulation or additional assumptions. Evolving model architectures by using certain fitness functions enables us to draw conclusions similar to those of evolutionary biologists based on real animals' anatomies. To our knowledge, a similar approach has only been tried by Olivares et al. [16] in recent years.

We take inspiration from the long tradition of using simple circuits and rules to model animal behavior in response to certain stimuli, both in real-life and simulated experiments. Grey Walter's work on the *Machina speculatrix* and the *Machina docilis* [24] showed that very simple electronic circuits can be sufficient to produce some behaviors reminiscent of innate animal behaviors and even classical conditioning. Note that the "learning circuits" were the result of an analysis of the operations "involved in establishing a connection between different stimuli to achieve a conditioned response". This means that they tell us something about the behaviors that can be achieved with simple circuitry, but not much about the way animal bodies and brains achieve similar behavior. Another example are Braitenberg vehicles. Initially conceived as a thought experiment, they are minimal robots whose wheel actuators are directly coupled to simple sensors. By varying the sensor-actuator connections, they can be made to show behaviors associated with living organisms, such as an enduring attraction or aversion to a light source, obstacle avoidance and chemotaxis [5,22].

This naturally leads to the question of how architecturally complex agents even need to be in order to show successful behaviors in their environment, including the social environment. Simulations of simple recurrent neural networks with two or three hidden nodes which control Braitenberg-like agents emitting acoustic signals have shown that social interaction itself can increase a network's complexity (as measured by the entropy between the internal nodes) and lead to interesting (though not necessarily lifelike) behavior patterns when two such agents interact [9,18,19].

To our knowledge, most of the work done in this area uses present-time sensory input and, if at all, introduces a temporal aspect through the use of recurrencies in the artificial neural network architecture. In Couzin et al.'s [11] zone model of fish social behavior, individual fish react almost instantaneously to others' movements. However, possible memory or prediction capabilities of

individual fish are not examined. On the other hand, Murakami, Niizato and Gunji [15] as well as Strömbom and Antia [23] model swarm dynamics using anticipation.

Given our interest in memory and anticipation of a social partner’s actions, we are looking for a way in which temporal dynamics (such as delayed or anticipatory reactions to a partner’s actions) may be detectable from an outside perspective in pre-recorded behavior data, without needing to experimentally manipulate the behavior. In addition, we want to find out what role such dynamics play in producing lifelike simulations of fish movements.

Based on the work reviewed so far, we decided to simulate fish interactions using minimal (at least at first) feedforward artificial neural networks and treating memory and prediction capacities as external “modules” rather than as properties of the neural architecture or internal model (as do, for example, Blum, Winfield and Hafner [4]. This is in line with Reynolds’ [20] approach of giving his artificial agents “approximately the same information that is available to a real animal as the end result of its perceptual and cognitive processes” (more details in the Methods section). This comes with the drawback of not being able to simulate two partners freely interacting: In order to have “predictions” of a partner, the focal agent model must interact with a pre-recorded one. However, this allowed us to directly compare the individual performance of models with the exact same partner and based on the exact memory/prediction timesteps they are receiving as input, while keeping their neural architecture simple.

The presence or absence of the temporal input modules were determined by an optimization process. In this process, the complexity of the neural network architecture (as measured by the number of hidden layers and their respective nodes) would be adapted in order to better approximate the “decision making function” mapping input states to motor actions. As target functions to optimize, we used two different measures in two separate experiments: In experiment 1, we used the framewise deviation of a model’s predictions from the original fish track it was trained on. In experiment 2, we used the “fish-likeness” of a model’s simulation, as rated by a long short-term memory (LSTM) discriminator trained for this purpose. Both were traded off against the number of trainable parameters in the models in order to encourage simplicity. By thus tweaking our modeling choices and examining the best fitting models, we hoped to answer the following questions: **Experiment 1:** Can we show in how far a given fish predicts or remembers another’s movements and changes its behavior accordingly? **Experiment 2:** What minimum of temporal input and internal complexity (taking into account Occam’s razor [6]) is sufficient to model a specific fish, or to produce generally “fish-like” behavior? In keeping with the origin of the biological agents we were modeling, we chose to use a custom Evolution Strategy [1] in order to optimize our model architectures while keeping them simple.

## 2 Methods

Please find a diagram of our workflow in Fig. 1c.

## 2.1 Ground Truth Data

Our efforts in both experiments were aimed at successfully modeling existing data from five live pairs of Trinidad guppies, filmed and tracked while moving freely in an experimental tank. Their movements were filmed in a white square tank of 88 cm width and 7.5 cm water depth (see Fig. 1a) and tracked at 30 frames per second (fps) using the BioTracker movement tracking software [14]. In every evolution run, one of the two fish per pair (the “focal fish”) was used as ground truth data to be modeled, while the other live fish’s trajectory functioned as the “social partner”.

## 2.2 Neural Network Models

We configured all models as multi-layer perceptrons using Keras [10]. Independently of the number of layers and the number of nodes in each layer, all hidden layers were densely connected and used leaky ReLU with an alpha of 0.01 as an activation function. The case of 0 hidden layers implemented a linear mapping of input to output (please see supplementary material for an illustration), with one output being clipped to minimum 0 (ReLU). All models were trained using the Keras adaptive moment estimation (“Adam”) optimizer with a clipping norm of 2.0, a maximum of 100 epochs but early stopping in the case of stagnant validation loss (with a patience parameter of 5). The two output nodes represented 1) the length/magnitude of the predicted movement, and 2) a change in heading direction in 2-argument arctangent ( $\text{atan2}$ ), respectively. The movement length node was a ReLU to prevent “negative” (backwards) movements. The direction change output node was not regularized. We used polar coordinates as this was computationally easier when using a field of view. Please see Fig. 1b for an example. If used for simulation, the predictions of the network were transformed into x and y displacement coordinates in the global cartesian coordinates. This was done by a custom function which also clipped the maximum length of the movement to 4cm to regularize the simulations and prevent “jumps” across the tank.

## 2.3 Input Information

We implemented the inputs to the network as higher-level “spatial awareness” modules: a wall detection module and a partner detection module. We decided to “outsource” the fish’s awareness of walls and partners in this way and give the model precise information in the same spatial format as the required output instead of, say, using simulated raycasting to detect walls and partners and letting the model combine the ray information into a spatial representation to act upon. This meant that a model only had to make the movement prediction, saving it some computation and eliminating one potential source of error. Input information was precise within a field of view of  $172^\circ$  on each side (which is realistic for this species of fish). For information on wall vision, please see the supplementary material. Figure 1b illustrates our input modules.

**Partner Detection Module (*Partner*).** The partner detection module computed the position of the partner fish from the simulated fish’s point of view and passed the polar coordinates as input to the neural network. It implemented the capability to “remember” or “predict” the partner fish’s position at other points in time. Given an integer  $-a$ , the module would compute the position of the partner fish  $a$  timesteps in the past relative to the *current* position of the focal fish, essentially allowing the model to remember past positions of the fish from its current perspective. Given an integer  $a$ , the module would compute the position of the partner fish  $a$  timesteps in the future (this information was available as our tracks were pre-recorded), allowing the model to predict the partner’s position from the perspective of the present. One model could use this module with more than one timestep at once depending on its hyperparameters, adding 2 input nodes (for the two coordinates) to the model per timestep used. The timesteps used by a given model were encoded as a list of integers in the hyperparameter “genome” which was subject to our evolutionary strategy.

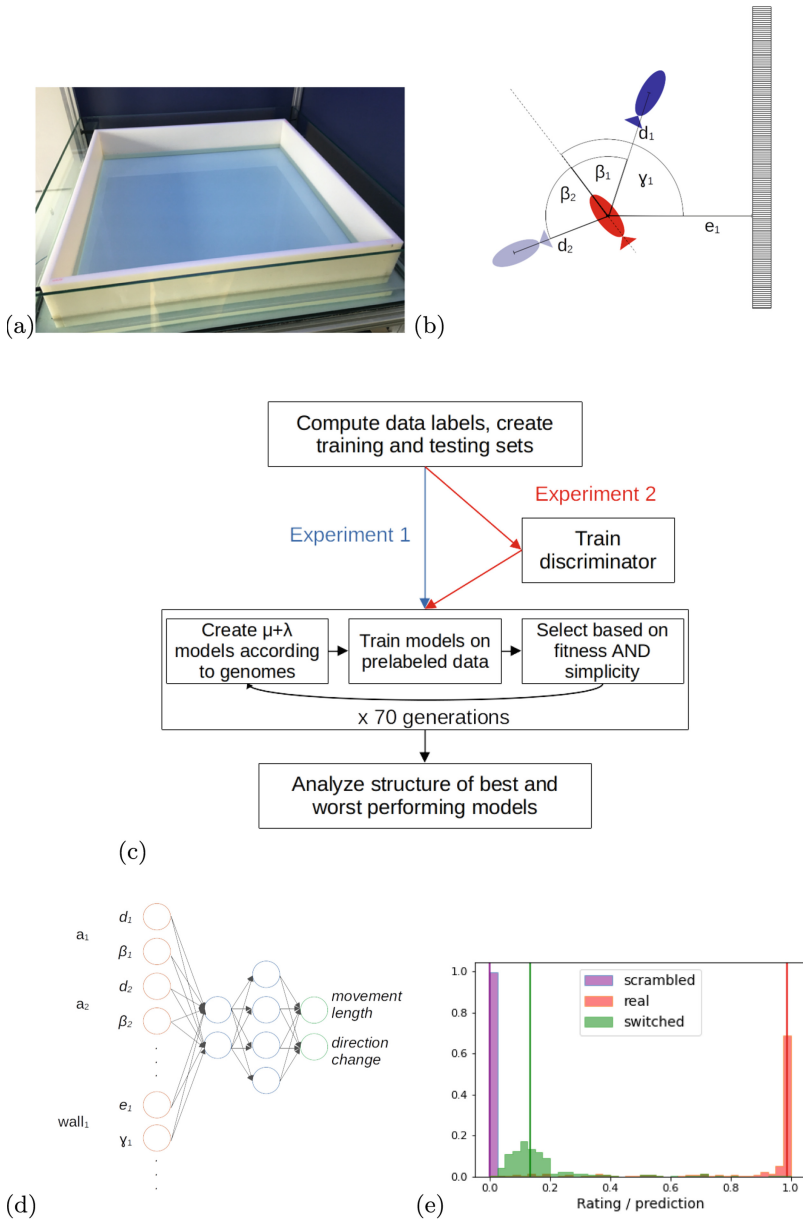
## 2.4 Evolution Strategy for Neural Network Architecture Search

Evolution strategies are a class of optimization methods first developed in the 1960s by Rechenberg and others [1]. Like other evolutionary computation methods, they use ways of “reproducing” and “selecting” artificial entities based on certain traits across multiple loops or “generations”, inspired by natural evolution. Unlike other methods, they usually evolve the entities’ mutation rates along with the other traits. The model hyperparameters subject to our evolution strategy were the following:

- The entirety of hidden layers and the number of nodes in each layer
- The partner perception timesteps used by the agent
- The presence or absence of wall input
- The magnitude of the noise added to all model inputs (more precisely, the factor by which the standard deviation of the entire input for a given feature was multiplied to form the standard deviation of a Gaussian with mean 0 to draw noise samples from)
- The noise level added to model outputs when simulating (more precisely, the factor by which the standard deviation of the original framewise fish movements was multiplied to form the standard deviation of a Gaussian with mean 0 to draw noise samples from)

These hyperparameters indirectly encoded the model architecture, and each model was trained using gradient descent with randomly initialized weights before testing it for selection. The outcomes we aimed to minimize simultaneously were a) the testing loss as measured in two different ways and b) the number of trainable parameters in the model (its complexity).

**Reproduction.** As is recommended for a combinatorial task [1], we used a  $(\mu + \lambda)$  evolution strategy, in which  $\mu$  parents are used to create  $\lambda$  offspring in



**Fig. 1.** a) Test tank for filming fish movements. b) Focal fish (red) and example input from partner fish (blue)  $d_1, d_2$  and  $\beta_1, \beta_2$ : Partner distances and angles for two example timesteps.  $e_1$  and  $\gamma_1$ : Wall distance and angle (only one wall shown). c) Workflow for both experiments. d) Example feedforward neural network (FNN) with several partner input timesteps, wall vision and hidden layers [2, 4]. e) Performance of lifelikeness discriminator: Relative frequency histogram of the ratings of the discriminator on unused real, scrambled and switched data. (Color figure online)

each generation, and the parents and offspring are then pooled and tested for selection. We did not use recombination to produce the offspring, but rather each selected parent was copied into the next generation (not the trained model but the hyperparameter “genome”), in addition to two mutated versions. Thus,  $\mu = 8$  and  $\lambda = 16$ . All copies and mutant versions of the selected parents were then built as feedforward neural networks (FNNs) and trained via gradient descent using pre-labeled data. Afterwards, they were tested for selection as described below. In the first generation,  $\mu = 8$  default models were initiated according to the settings in Table 1, and two mutant versions created for each to form the initial population. The mutation itself was governed by two “strategy parameters” (mutation rate and strength) which were part of the model genome, and evolved along with it.

**Mutation.** Mutation was carried out as follows. In addition to the model hyperparameters (or “object parameters” in ES speak), we evolved two “endogenous strategy parameters” of each model: its mutation strength  $s$  and its mutation rate  $p$ . The mutation strength was a factor applied to mutation steps in real-valued model parameters, and the mutation rate used as  $p$  for sampling from a Bernoulli distribution to determine whether a binary mutation would take place or not. For creating each mutant version of a parent, we randomly increased or decreased  $s$  by 1 (and then clipped it at a minimum of 1), randomly increased or decreased  $p$  by 0.1 (clipped to stay between 0.1 and 1), flipped wall input on/off with probability  $p$ , added a new partner perception timestep randomly chosen from  $-150,150$  with probability  $p$ , randomly subtracted or added  $s$  to a randomly chosen existing partner time step, removed one randomly chosen partner timestep with probability  $p$ , randomly increased or decreased the input noise factor by  $s/20$ , randomly increased or decreased the motor noise factor by  $s/20$ , added a hidden layer with  $s + 1$  nodes with probability  $p$ , added a node to a randomly chosen hidden layer with probability  $p$ , and removed one randomly chosen hidden layer with probability  $p$  (provided there was one). In order to avoid duplicate partner timesteps, the set of partner timesteps was used after mutation.

**Table 1.** Hyperparameters of default model.

Hyperparameter	Value
Partner timesteps	[0]
Wall vision	False
Input noise factor	0.0
Motor noise factor	0.0
Hidden layers	[0]
Mutation rate	0.3
Mutation strength	1

**Selection.** We performed two neuroevolution experiments, using two different forms of testing loss according to the different study aims. In both methods, the testing loss was traded off against the number of model parameters to select models for reproduction. All our datasets were used in both experiments. In **experiment 1**, we used the loss from testing each model on pre-labeled, unseen data from the same dataset. This served to gauge in how far the models were able to approximate the framewise movement “decisions” of the original fish, as the model received input based on the original fish’s real positions. The model architectures evolved by this method served to illuminate the individual behavior of the respective fish, and its unique dynamics with the partner fish. For each dataset, we did one evolution run using the first fish as the focal fish, and another using the second fish. In **experiment 2**, we used the negative ratings of the discriminator model described in Sect. 2.8 as the testing loss. Only the first fish of each dataset was used here. Each model was still built and trained on pre-labeled data in the same way as in experiment 1, but then made to freely simulate 400 frames of fish track given the original partner input, to be rated by the discriminator. This served to select for models what were able to produce the most fish-like trajectories and interactions with the partner fish, thus giving us more general insights about the behavior of these guppies.

In both experiments, at each generation of an evolution run we computed the Pareto optimal model genomes according to both minimal testing loss and minimal number of trainable parameters in the model. A (strongly) Pareto optimal data point in a two-dimensional feature space is one where, if there exists another data point with a better value on one feature, that data point must rate worse on the other feature [13]. This allowed us, in the first round, to select models according to both our desired outcomes (low testing loss *and* low complexity) without having to weigh them against each other. However, as the number of Pareto optimal models (also called the Pareto front) is not necessarily  $= \mu$ , we used the average of the min-max scaled test loss and complexity for each model as a secondary criterion for ranking the models, and then used the  $\mu$  best models for reproduction, making this an elitist selection technique. The procedure was repeated for 70 generations per run.

## 2.5 Data Labels

We trained all models in both experiments using labels based on the ground truth data. The labels represent the direction change of the original fish in radians and its movement length in cm at each frame, with the mean squared error of both output nodes as the loss function for each one. As the two outputs were roughly on the same scale (movement length in cm and direction change in radians were both expected to be in  $[0, 1]$ ), we weighted the output nodes equally for the final model loss. Training loss was defined as the deviation of the predicted move from the ground truth of the recorded focal fish. While this is obviously not biologically realistic as no animal has a deterministic trajectory to follow, our whole study rests on the assumption that a fish’s framewise movements can be



at least partly predicted based on its perception of the partner’s position and the tank walls.

## 2.6 Simulation

Each trained model could be used to simulate a fish trajectory given the partner and wall input suitable to its input configuration. Together with the partner trajectory, this could then be passed to a pre-trained discriminator model to be rated for “lifelikeness”. The framewise outputs of the simulator model would be transformed into movement coordinates. A movement was only performed if the resulting fish position was inside the tank walls of the dataset currently used; if not, the simulated focal agent simply remained in the same place but the simulation continued, feeding it new input for new potential movements.

## 2.7 Analysis of Evolution Results

As it was difficult in our application to verify the correct running of the Evolution Strategy objectively, we relied on the testing loss development across generations. Runs in which testing loss did not sink towards the final generations, did not reach levels below 0.5 or oscillated greatly throughout were regarded as “not converged”. However, by analyzing the best and worst performing models across all generations, we can still glean some cautious insights into the solutions found by these runs. Comparing the best models for the two fish in each dataset allowed us to draw inferences about the dynamic between the two fish. What all the evolution runs seemed to have in common was a marked bimodal distribution of testing loss across generations and model hyperparameters (see plots in the supplementary material). Given this pattern in the testing loss, we decided to compare the ensembles of the 30 best performing and the 30 worst performing models for each fish with two-sample t tests to gain information on the hyperparameter choices for modeling that fish. Comparing the partner timestep histograms between the best and worst models for each fish provides information on the partner timestep input likely used by that fish. While experiment 2 was more exploratory, we formulated specific predictions for experiment 1: If both fish in a pair mostly ignored each other, we would expect the timestep histograms to be inconclusive for both fish respectively, as the evolution strategy would not find an adequate model of either fish’s movements based on the other one. If one fish in a pair mostly remembered or anticipated the other’s movements but was itself largely ignored, then the timestep histograms for the first fish should show a predominance of the best models within a certain time range. The other, “careless” fish in this scenario should either have inconclusive timestep histograms, or a predominance of timesteps exactly the opposite of the first fish. If both fish coordinated their movements closely, with one mostly reacting and the other mostly anticipating, we would also expect a “mirrored” pattern in their timesteps. Such a mirrored pattern would therefore be difficult to interpret causally, while the other patterns would present a clear picture.

## 2.8 Simulation Discriminator

In order to be able to automatically evaluate the lifelikeness of a simulated fish pair trajectory, we built a discriminator LSTM model trained to distinguish real pre-recorded fish pairs from a) datasets with scrambled frames (i.e. completely un-fish-like trajectories) and b) real focal and partner fish trajectories switched between different datasets (i.e. fish-like but not interacting). It was loosely inspired by the discriminator part of Gupta et al.’s [12] Social GAN for producing socially acceptable walking trajectories. The discriminator contained one densely connected layer of 64 LSTM cells and one output node regularized with a sigmoid function. We trained using binary cross entropy loss and the “Adam” optimizer. The discriminator was trained on 35 datasets of recorded fish pairs not used in the later study. When testing on unused cuts of the training data sets, the discriminator did a good job separating the real fish pair data from scrambled and switched tracks (see Fig. 1e). For additional information, please see the supplementary material.

**Code Availability.** All our code is available at [gitlab.com/leamusi/fish\\_simulation](https://gitlab.com/leamusi/fish_simulation) (main project code) and [gitlab.com/leamusi/fish\\_movements](https://gitlab.com/leamusi/fish_movements) (additional tools for processing fish movements).

## 3 Results

### 3.1 Experiment 1: Selection Through Testing on Pre-labeled Data

For each dataset of two fish interacting, we did one evolution run using the first fish as the focal fish, and another using the second fish. Please find detailed results and plots in the supplementary material. For all runs, the majority (at least 60%, usually more) of the best performing models had no wall vision. Results on motor noise are not reported when testing on labeled data, as they only come to bear when testing in simulation. For dataset N07P3, convergence was unclear while evolving models for either fish. Partner timesteps 50 frames in the future were advantageous for modeling the first fish, while input 50 frames from the past helped the best models of the second fish. For N12P4, the evolution processes for both fish achieved solutions with comparatively low testing loss. The best models of the first fish were dominated by inputs 50–100 frames or 1.6–3.3 s in the past, while the temporal inputs for the second fish were inconclusive. For dataset N13P3, it appears that only the evolution process for modeling the first fish converged on good solutions. The timestep results for both fish were inconclusive. In dataset N13P4, only the evolution strategy for the first fish achieved solutions with low loss. The first fish’s temporal input was dominated by information 0 to 100 frames in the future, while the second fish’s was dominated by past input (100 to 0 frames back). For dataset N16P4, convergence of the evolution process was unclear for the first fish, but achieved good solutions for the second one. The results regarding temporal input were inconclusive for both fish.

T tests on the other hyperparameters between the best and worst performing models were generally nonsignificant except for input noise factor (which was always lower in the best models) and the number of hidden nodes (which was significantly higher in the best models for several fish).

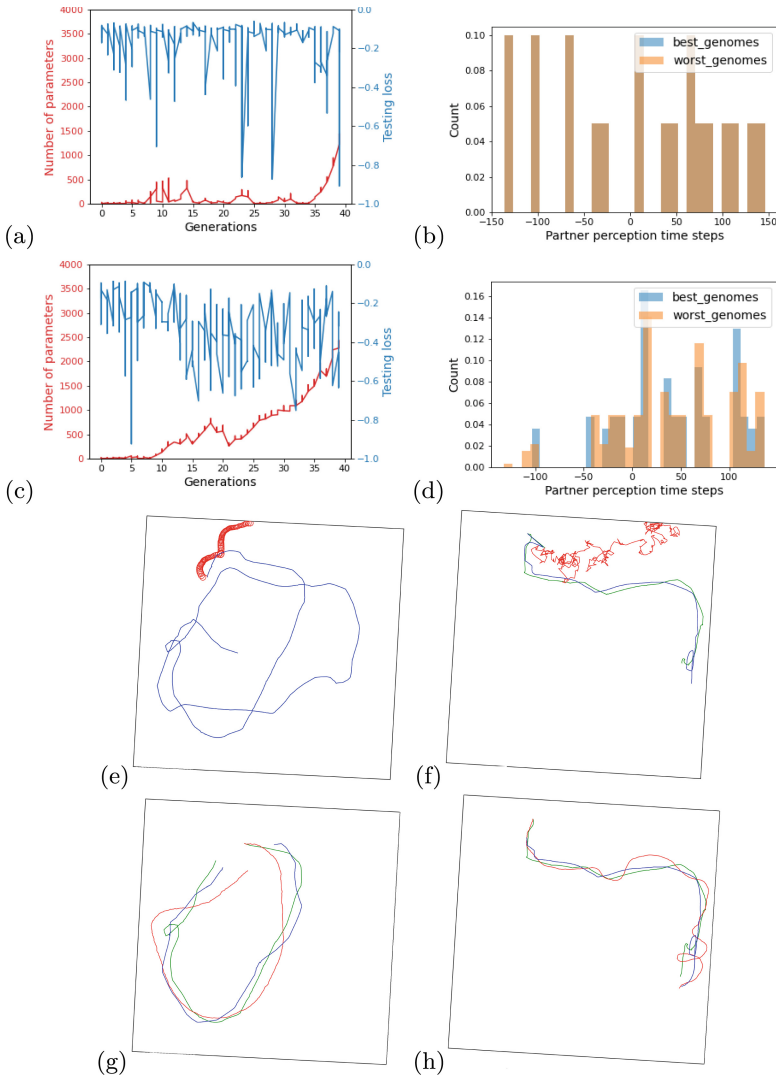
### 3.2 Experiment 2: Selection Via Discriminator Ratings

**Fixed Strategy Parameters.** The performance of the evolution strategy using discriminator ratings fell short of our expectations as we had reached ratings of up to 0.9 in a pilot study with 40 generations and *fixed* strategy parameters: the mutation strength was set at 1 and the mutation rate at 0.3. We therefore report a selection of these results here (Fig. 2a–d), including the resulting simulations (see Fig. 2g–h). For dataset N07P3, there was no significant difference on any hyperparameter except input noise factor ( $\delta = -0.09, t(58) = 2.4, p < 0.05$ ), while both the best and the worst models had an average of 31 hidden nodes and 20 partner timesteps. Timestep distributions of the best and worst models were identical. For dataset N12P4, there was no significant difference on any variable, no clear picture in the timestep distribution, and some of the best rated models had 0 hidden nodes and a minimum of 7 partner timesteps. The best models for the other datasets only achieved top ratings of 0.4. For the results obtained using evolving strategy parameters, please see the supplementary material.

## 4 Discussion

### 4.1 Experiment 1

The bimodal distribution of the testing loss seems to indicate that for each dataset, there are two attractors for a model’s testing loss to gravitate towards, with the lower one being a natural baseline loss. This baseline may be due to the fact that a) there may be systematic input factors or modeling choices which our study does not account for or b) no focal fish’s behavior is entirely deterministic, and therefore no model can be expected to recreate it perfectly. The absence of wall vision in most best performing models may be easily explained by the fact that partner fish (like focal fish) usually stayed close to the walls, meaning that information about the walls was contained in partner input, and wall vision input would thus have been redundant. The partner timestep results for both fish in dataset N07P3 clearly mirror each other. This makes it difficult to state whether one fish was anticipating the other, the second fish was following the first reactively, or both. However, it is clear that the two fish’s movements have a strong temporal connection. The results for N12P4 suggest that the first fish appears to have used its memory to follow its partner, while the partner itself was not minding the first fish at all. In N13P4, the temporal results again mirror each other, making causal inference difficult but showing a clear temporal link



**Fig. 2. a)–d):** Results of neuroevolution using selection via discriminator with fixed strategy parameters. Left column: Change of testing loss (blue) and model complexity (trainable parameters, red) across evolution generations. Right column: Histograms of partner input timesteps for the 30 best (blue) and the 30 worst (orange) models. a)–b) Dataset N07P3. c)–d) N12P4. **e)–h):** Simulations by different models: original focal fish in green, partner in blue, simulation in red. e) Simulation by one of the best 30 models for the first fish of dataset N07P3 evolved using prelabeled data. f) Simulation by the best model for the dataset N12P4 evolved using the discriminator, with evolving strategy parameters. Discriminator rating 0.5. g) Simulation by the best model for dataset N07P3 evolved using the discriminator, with fixed strategy parameters. Discriminator rating 0.91. h) Simulation by the best model for dataset N12P4 evolved using the discriminator, with fixed strategy parameters. Discriminator rating 0.92. (Color figure online)

in the pair’s behavior. From the lack of convergence and inconclusive histograms for the two fish pairs in datasets N13P3 and N16P4, it appears that the two partners did not mind each other much in either case, which is reflected in a visualization of the raw data (see supplementary material, Fig. 1).

The results also indicate that models seem to benefit from a low input noise factor when tested on labeled data, although the average input noise factor in the best models was still nonzero for all datasets. This suggests a sweet spot for input noise, at which the noise increases model robustness but does not distort predictions too much. Apart from this, the results on the general hyperparameters suggest an advantage of having more hidden nodes for modeling some datasets. Taken together, these results show how evolving the architecture and input configuration of an FNN can be used to infer the interaction dynamics of fish pairs. It must be remarked that even the best models evolved with this selection method did not produce very realistic-looking simulations (see Fig. 2e). This is likely due to the fact that training and testing were done on input computed frame-wise from the perspective of the pre-recorded focal fish. When simulating freely, the agent may move into positions relative to the partner fish that would be unusual for the real focal fish, meaning that the model did not have “experience” with such input, and was not selected or trained for it.

## 4.2 Experiment 2

Using the simulation discriminator for selection in the evolution strategy means that the results cannot tell us much about the individual datasets being modeled. The discriminator rates the partner trajectory together with the simulated focal fish on how much they resemble a real fish pair, but we cannot say for certain what this rating captures, or what influence the specific partner trajectory has. Therefore, the best performing genomes can teach us something about creating lifelike fish simulations in general, but not about imitating a specific focal fish. We therefore interpret the results jointly, without focusing on each specific dataset.

*With Fixed Strategy Parameters.* The similarity of the best and worst model architectures is an astonishing result. It indicates that the only thing which made a difference here was the random weight initiation when compiling the models, which led some to learn very high quality simulation skills, while others possibly got caught in a local minimum. This leads us to the conclusion that a suitable model architecture is a necessary but not a sufficient condition for good model performance, and that the contribution of the model training should not be underestimated. While in experiment 1 specific timesteps and low input noise were advantageous for modeling specific fish, it appears that something else is necessary (but not sufficient) when trying to fool a discriminator: the number and variety of partner timesteps. Given this, hidden nodes may not even be necessary, and a very simple function of the partner input may produce highly lifelike simulations.

## 5 Summary

We show that by evolving model architectures and input configurations in experiment 1, we can capture how some guppies' movements can be predicted through their memory and others' through their anticipation of a partner's movements, with the input-output relationship flexibly determined by an FNN as a function approximator. In general, more hidden nodes in an FNN did not necessarily seem to bring an advantage in all pair interactions: the behavior rules leading from partner perception to the fish's own behavior can be very simple, confirming the findings of previous models of fish behavior. For future studies, being able to infer the temporal dynamics between two fish without experimental manipulation means that we have a new, more efficient way of gauging aspects of a fish's social personality (does it tend to react or anticipate?). When attempting to build socially competent fish robots, we can then adapt their behavior to what we already know about the specific social partner. Just as in experiment 1, the results of experiment 2 show that very simple fish behavior models can produce lifelike simulations. We also learn from this experiment that the architecture itself does not determine model performance: rather, the right architecture seems to be necessary but training is key.

## 6 Limitations and Outlook

One obvious limitation of our study is the fact that our models do not guarantee causality: finding out that a given FNN approximates the function connecting hypothetical visual input received by a fish to its movements does not mean that the real fish was actually using such inputs and performing such computations. In the case of partner input from the future, for example, it is theoretically possible that rather than the focal fish predicting the partner's future position, the partner is systematically positioning itself a certain way in relation to where it saw the focal fish looking seconds ago. However, the benefits of such behavior would be unclear. Another limitation of our approach should be addressed, as it arises naturally from the work we reviewed above: Our models are clearly not embodied (not even within the simulation), and we do not account for the physical dynamics of fish movements at all. This means that our study is only a first step towards building accurate models of fish movement, which also happens to answer some more questions about the original data. In future, our models could undergo further evolution when combined with a model of fish's bodies and movement dynamics. An example for this could be the models of burst-coast swimming dynamics presented by Calovi et al. [8] and Sbraraglia et al. [21]. It is to be expected that such a joint evolution of social and physical movement dynamics may produce different solutions than this study, and produce new insights.

## References

1. Beyer, H.G., Schwefel, H.P.: Evolution strategies - a comprehensive introduction. *Nat. Comput.* **1**(1), 3–52 (2002). <https://doi.org/10.1023/A:1015059928466>
2. Bierbach, D., Gómez-Nava, L., Francisco, F.A., Lukas, J., Musiolek, L., Hafner, V.V., Landgraf, T., Romanczuk, P., Krause, J.: Live fish learn to anticipate the movement of a fish-like robot. *Bioinspiration Biomimetics* **17**(6), 065007 (2022). <https://doi.org/10.1088/1748-3190/ac8e3e>
3. Bierbach, D., Sassmannshausen, V., Streit, B., Arias-Rodriguez, L., Plath, M.: Females prefer males with superior fighting abilities but avoid sexually harassing winners when eavesdropping on male fights. *Behav. Ecol. Sociobiol.* **67**(4), 675–683 (2013). <https://doi.org/10.1007/s00265-013-1487-8>
4. Blum, C., Winfield, A.F.T., Hafner, V.V.: Simulation-Based Internal Models for Safer Robots. *Frontiers in Robotics and AI* **4** (2018). <https://doi.org/10.3389/frobt.2017.00074>
5. Braitenberg, V.: *Vehikel: Experimente mit künstlichen Wesen*. LIT Verlag Münster (2004)
6. Britannica, E.: Occam's razor | Origin, Examples, & Facts | Britannica, August 2023. <https://www.britannica.com/topic/Occams-razor>
7. Brown, C., Laland, K.N.: Social learning in fishes: a review. *Fish Fish.* **4**(3), 280–288 (2003). <https://doi.org/10.1046/j.1467-2979.2003.00122.x>
8. Calovi, D.S., Litchinko, A., Lecheval, V., Lopez, U., Escudero, A.P., Chaté, H., Sire, C., Theraulaz, G.: Disentangling and modeling interactions in fish with burst-and-coast swimming reveal distinct alignment and attraction behaviors. *PLoS Comput. Biol.* **14**(1), e1005933 (2018). <https://doi.org/10.1371/journal.pcbi.1005933>
9. Candadai, M., Setzler, M., Izquierdo, E.J., Froese, T.: Embodied dyadic interaction increases complexity of neural dynamics: a minimal agent-based simulation model. *Front. Psychol.* **10**, 540 (2019). <https://doi.org/10.3389/fpsyg.2019.00540>
10. Chollet, F.: Keras (2015). <https://keras.io/>
11. Couzin, I.D., Krause, J., James, R., Ruxton, G., Franks, N.R.: Collective memory and spatial sorting in animal groups. *J. Theor. Biol.* **218**(1), 1–11 (2002). <https://doi.org/10.1006/jtbi.2002.3065>
12. Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., Alahi, A.: Social GAN: Socially Acceptable Trajectories with Generative Adversarial Networks, March 2018. <https://doi.org/10.48550/arXiv.1803.10892>
13. Mock, W.B.T.: Pareto optimality. In: Chatterjee, D.K. (ed.) *Encyclopedia of Global Justice*, pp. 808–809. Springer, Netherlands, Dordrecht (2011). [https://doi.org/10.1007/978-1-4020-9160-5\\_341](https://doi.org/10.1007/978-1-4020-9160-5_341)
14. Mönck, H.J., et al.: BioTracker: an open-source computer vision framework for visual animal tracking, March 2018. <https://doi.org/10.48550/arXiv.1803.07985>
15. Murakami, H., Niizato, T., Gunji, Y.P.: Emergence of a coherent and cohesive swarm based on mutual anticipation. *Sci. Rep.* **7**(1), 46447 (2017). <https://doi.org/10.1038/srep46447>
16. Olivares, E., Izquierdo, E.J., Beer, R.D.: A Neuromechanical Model of Multiple Network Rhythmic Pattern Generators for Forward Locomotion in *C. elegans*. *Frontiers in Computational Neuroscience* **15** (2021)
17. Pezzulo, G., Butz, M.V., Castelfranchi, C.: The anticipatory approach: definitions and taxonomies. In: Pezzulo, G., Butz, M.V., Castelfranchi, C., Falcone, R. (eds.) *The Challenge of Anticipation*. LNCS (LNAI), vol. 5225, pp. 23–43. Springer, Heidelberg (2008). [https://doi.org/10.1007/978-3-540-87702-8\\_2](https://doi.org/10.1007/978-3-540-87702-8_2)

18. Reséndiz-Benhumea, G.M., Sangati, E., Froese, T.: Levels of coupling in dyadic interaction: an analysis of neural and behavioral complexity. In: 2020 IEEE Symposium Series on Computational Intelligence (SSCI), pp. 2250–2256, December 2020. <https://doi.org/10.1109/SSCI47803.2020.9308429>
19. Reséndiz-Benhumea, G.M., Sangati, E., Sangati, F., Keshmiri, S., Froese, T.: Shrunk social brains? a minimal model of the role of social interaction in neural complexity. *Frontiers in Neurobotics* 15 (2021)
20. Reynolds, C.W.: Flocks, herds and schools: a distributed behavioral model. *ACM SIGGRAPH Comput. Graph.* **21**(4), 25–34 (1987). <https://doi.org/10.1145/37402.37406>
21. Sbragaglia, V., Klamser, P.P., Romanczuk, P., Arlinghaus, R.: Evolutionary Impact of Size-Selective Harvesting on Shoaling Behavior: Individual-Level Mechanisms and Possible Consequences for Natural and Fishing Mortality. *Am. Nat.* **199**(4), 480–495 (2022). <https://doi.org/10.1086/718591>
22. Shaikh, D., Rañó, I.: Braitenberg Vehicles as Computational Tools for Research in Neuroscience. *Frontiers in Bioengineering and Biotechnology* 8 (2020)
23. Strömbom, D., Antia, A.: Anticipation induces polarized collective motion in attraction based models. *Northeast J. Complex Syst.* **3**(1), March 2021. <https://doi.org/10.22191/nejcs/vol3/iss1/2>
24. Walter, W.G.: A machine that learns. *Sci. Am.* **185**(2), 60–64 (1951)