UNIVERSITÀ DEGLI STUDI DI TRIESTE

XXXIII CICLO DEL DOTTORATO DI RICERCA IN

INGEGNERIA INDUSTRIALE E DELL'INFORMAZIONE

Artificial Intelligence Strategies in Multi-agent Reinforcement Learning and Robotic Agents Evolution

Settore scientifico-disciplinare: **ING-INF/05 SISTEMI DI ELABORAZIONE DELLE INFORMAZIONI**

DOTTORANDO / A
Jacopo Talamini

COORDINATORE
Prof. Fulvio bBabich

SUPERVISORE DI TESI
Prof. Eric Medvet

ANNO ACCADEMICO 2019/2020

UNIVERSITÀ DEGLI STUDI DI TRIESTE

DOCTORAL THESIS

Artificial Intelligence Strategies in Multi-agent Reinforcement Learning and Robotic Agents Evolution

Author:
Jacopo TALAMINI

Supervisor:
Dr. Eric MEDVET

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in

Information and Industrial Engineering

November 28, 2020

"Without deviation from the norm, progress is not possible."

Frank Zappa

Abstract

Most of the theoretical foundations which have contributed to shape Artificial Intelligence (AI) as we know it come from the last century. The technological advancement of the last decades however, mainly in the form of faster parallel computation, larger memory units, and Big Data, has dramatically increased the popularity of AI within the research community.

Far from being only a pure object of research, AI has been successful in many fields of applications, and it has become deeply integrated into our daily experiences. We live in a society in which on-demand content suggestions are tailored for each customer, where it is possible to order products online by chatting with bots. Smart devices adapts to the owner behavior, the stock exchange brokers are algorithm based on predictive models, and the computers are able to discover new medicines and new materials.

Despite the amount of knowledge acquired on AI, there are still many aspects of it that we do not fully understand, such as the interplays within multiple autonomous agents scenarios, in which AIs learn and interact in a shared environment, while possibly being subjected to different goals. In these scenarios the communication and the regulation of the autonomous agents are both extremely relevant aspects.

In this work we analyze in which way the language expressiveness affect how agents learn to communicate, to which extent the learned communication is affected by the scenario, and how to allow them to learn the optimal one. We then investigate which communication strategies might be developed in different scenarios when driven by the individual goal, which might lead to improved equality in a cooperative scenario, or more inequality in a competitive one. Another aspect that we consider is the ethics of multiple agents, to which we contribute by proposing a way to discourage unethical behaviors without disabling them, but instead enforcing a set of flexible rules to guide the agents learning.

AI success can be determined by its ability to adapt, which is an aspect that we consider in this work, relatively to the adaptation of autonomous soft robotic agents. Soft robots are a new generation of nature-inspired robots more versatile and adaptable than the ones made of rigid joints, but the design and the control of soft robots can not be easily done manually. To this extent we investigate the possibility of mimicking the evolution of biological beings, by adopting evolutionary meta-heuristics for optimizing these robots. Specifically we propose to evolve a control algorithm that leverages the body complexity inherent to the soft robots through sensory data collected from the environment. Considering the problem of designing adaptable soft robots, we propose an approach that allows to automatically synthesize robotic agents for solving different tasks, without needing to know them in advance.

Agent-based scenarios are powerful research tools that can be adopted also for approximating the behavior of biological actors. Based on this possibility, we propose a model for the assessment of the publishing system indicators, which are currently used to evaluate authors and journals.

Acknowledgements

Foremost, I would like to express my gratitude to my Ph.D. supervisor Eric Medvet, for believing in me, and for being the one I could always count on for answering my questions and doubts. Eric has been responsible for creating the research lab in which most of my work took place. This is called the Evolutionary Robotics and Artificial Life Lab, a research environment where he always encouraged and stimulated original thinking and creativity.

A special thank goes to Stefano Nichele and all the people of the Living Technology Lab at the Oslo Metropolitan University, for allowing me to spend this last period of my Ph.D. in Norway, and having the chance to collaborate with such a talented and enthusiastic team of researchers.

I would like to thank Lars Kjemphol, artist, writer and good friend of mine, who gave precious tips and feedback to improve the quality of this document.

It would have been difficult to complete this academic path without the support and the confrontations with my best friends (and accidentally also fellow engineers). I feel grateful to all of them, but in particular the ones that have been closer to me in the last years: Vladyslav (Vlady), Emanuele (Vez), Massimo, Giovanni, and Luca (Bat).

Needless to say, I am extremely thankful to my parents Lorella & Pieraugusto, to my younger brother Nicolò (Nano), and to my good dog Cosmo. Thank you for being my family and for always respecting and supporting my decisions.

Finally, all my gratitude goes to my girlfriend Cristina, who has been on my side all these years. Without her constant support and patience this achievement would not have been possible, and I would not be the same person I am now.

This is dedicated to all of you.

Contents

Summary

In the early days of AI, computers were used to solve tasks considered difficult by humans, or too time consuming for them. In order to be understood by the computers, this kind of tasks required to be carefully expressed in the form of algorithms, defined through mathematical rules that are complex for us to understand.

On the other side, the computers struggled at accomplishing tasks which formulation is immediately understandable by humans, and which require reasoning and decision making skills. This second kind of tasks, on the contrary, are difficult to express in the form of algorithms and could be very heterogeneous, but they all require some kind of perception and cognitive capabilities. It is precisely within this context, that AI was considered promising at overcoming the human limits, by proposing mathematical abstractions inspired by the cognitive features of the biological brain.

Despite the development of many AI techniques, for a long time it was not possible to use them to solve these kind of tasks, since both their formalization and the optimization were too computationally demanding for the resources available. During the last century, these limitations led to periods of reduced funds and interest among the researchers, also known as the AI winters. This issue seems to be partially overcome during the past decades, thanks to the technological advancement in the manufacturing of Graphic Processing Units (GPUs) for optimized parallel computation, to the availability of larger memories which have allowed to operate with bigger collections of data, and finally thanks to the digitalization of society and Big Data, which has allowed to collect huge collection of data to make the machines learn from.

In this work we employ AI for solving problems that require cognitive capabilities, where the goal of the AI is to find an optimal strategy, without any prior knowledge on the task, provided only with an environment in which it can make decisions and observe the consequences. Approaching these cognitive problems using AI is extremely relevant because, on one side it does not require the engineers or researchers to manually design the algorithm corresponding to the optimal strategy, and on the other side because the AI might discover a different solution to a task w.r.t. the ones that we expect.

Reinforcement Learning (RL) is an AI approach that allows machines to learn decision making by directly interacting with the environment. According to RL, an artificial agent improves its skills through trial and error, guided only by a reward signal, in a way that is inspired by how animals learn. The goal of the RL agent is therefore to maximize not the immediate reward, but instead the cumulative reward collected after a series of interactions with the environment. For this reason, in such a long term perspective, a decision leading to an higher cumulative reward might be preferred over an immediately higher rewarding one.

We must notice that RL is not a form of Supervised Learning: no label for the correct decision for the current input is given, but instead the machine updates its strategy guided only by the maximization of its cumulative reward throughout all the possible trajectories of sensory inputs and corresponding actions.

According to RL, the agent is not aware of the underlying physical laws of the environment, and typically it does not even know the complete state in which the environment is at a certain time, but rather it senses only part of it.

The environment in RL is typically expressed in the form of a Markov Decision Process (MDP), according to which all the possible states are the nodes of a graph, connected by directed edges which represents the action performed by the agent, the state transition probability, and a reward consequent to the transition from the previous state. In a MDP it holds the Markov property, which guarantees that the

state transition probability is influenced only by the current state of the environment, and by the action taken in that state, while all the previous states and actions are irrelevant for that transition.

The formalization of the RL problem requires an MDP which defines the environment, a reward function, and a policy function. Specifically, the reward signal is responsible for driving the agent strategy to the optimal one, and since the reward signal is problem-dependent, its design can be challenging, since there might be many solutions for a specific task. The strategy of an agent in RL is represented by a policy function which maps each observation to an action. A policy is optimal if an agent, following that policy, is able to collect the maximum cumulative reward over a trajectory of states and actions.

RL has showed promising results in wide fields of decision making that are worth to mention, such as playing Atari video games at human-level through visual inputs. Games are extremely relevant as AI benchmarks, since they can be viewed as simulation of real life tasks, on which we can measure the AI performance, and then we can compare them with other AIs and humans.

More recently RL was able to beat Go world champion, without any prior knowledge of the game, and being limited to observing the chessboard. Analyzing the matches in which the AI played against the world champion, it has been possible for Go masters to learn from the machine successful strategies that were previously underestimated.

Another direction of AI research is using this framework for solving problems that are challenging and not intuitive for humans, which typically require to find a solution in an extremely high-dimensional space. One of these problems is the adaptation of robotic agents, a possibly large scale process that allows to assess the ability of these robots to succeed in a given environment. Seeking to bridge the gap between machines and living beings, these nature-inspired robots are designed in such a way to exhibit almost infinite degrees of freedom, thus allowing them to interact with humans in a safe way, and making them capable of performing movements and interactions with the environment inspired by the way plants and animals do. Despite being free from most of the constraints that affect traditional robots, soft robots are extremely difficult to design and control using traditional robotics techniques. Several researches for this reason have suggested that these robots design and control should be done by means of meta-heuristics.

Evolutionary Computation (EC) is an optimization algorithm inspired by the biological evolution. EC allows to find optimal solutions by modeling the search within the solution space through a dynamic populations of individuals that compete for limited resources, where only the fittest individuals survive and reproduce, and where the offspring inherit some traits from the parents. The population dynamics in EC occurs in a form inspired by the natural selection, where the new individuals are generated from the initial ones by means of the genetic operators, which allow to implement the trade-off exploration-exploitation.

This optimization approach need no prior on the nature of the solution, nor on the fitness landscape, but instead requires the researcher to develop a way to assess candidate solutions, and a mapping function from the EC internal representation of the solutions to its actual form.

Throughout this document we present a number of contributions specific to the communication and the regulation in multi-agent systems, and about the adaptation of robotic agents. Here we briefly list the contributions of this work:

i We design a multi-agent cooperative game, in which the autonomous agents learn to develop a simple form of communication in order to tell necessary information to succeed at the game. In this scenario we investigate the impact of the expressiveness of the language used for the communication, both in terms of the efficiency and effectiveness of the learned strategies. The results show how the language expressiveness should be designed in a given scenario to allow both the highest effectiveness and efficiency.

ii We consider three different scenarios for a resources collection game, in which a form of broadcast communication is necessary for performing optimally. In this game we investigate the impact of communication on the individual strategies that are developed by the agents in the different scenarios, where each scenario is defined by a different relationship individual-collective goal. The results show that in the cooperative scenario the learned communication strategy lead to an evenly distribution of the resources among the agents, while in the competitive scenario each agent develops a selfish strategy which favors the inequality.

iii Motivated by the lack of regulations for self-driving cars, we consider the problem of enforcing traffic rules on agent-based autonomous vehicles as part of the learning process. To tackle this problem we design a simple road traffic simulator, in which each driver is able to update the speed and the currently-occupied lane of its vehicle. The traffic rules are expressed in the form of a negative feedback assigned to the drivers for breaking the corresponding rule, and thus the agents are trained with these additional rewards. The experimental results show that such rules result into a safer, but slightly less efficient traffic flow, w.r.t. the one resulting without rules. The most relevant aspect of this approach however is that the compliance to these relies only on the optimal policy learned, meaning that these rules might be evaded in favor of a greater good.

iv We consider the problem of finding the optimal control law for robotic agents in the form voxel-based soft-robots. These extremely versatile robots are made of flexible building blocks called voxels, which can expand or contract when controlled by an external signal, and are motivated by the idea of shifting the robot complexity from the controller to the body. We explore the possibility of taking advantage of this paradigm by providing these robots with distributed sensing, and a controller based on an artificial neural network. We compare this control approach with another non-sensing one from the literature, by manually crafting a series of robotic bodies on which we evaluate both controllers, by optimizing each one on locomotion tasks. The experimental results show that the sensing controller proposed in this work is a promising approach, which outperforms the non-sensing counterpart, despite computationally expensive to optimize.

v We focus on the problem of automatically discover adaptable soft-robots bodies that allow to succeed at many different tasks. To solve this problem, we define a measure of adaptability corresponding to the self-organized criticality condition, which is proper of the dynamical systems, and that allows the most adaptable and complex behaviors. A number of robotic bodies are evolved towards this condition, and we then compare the resulting bodies on three tasks, against several other bodies, some of them inspired by the ones from the previous work, as well as some others randomly generated. We conclude showing that our approach for estimating adaptability allows to evolve bodies that are versatile, and successful at different tasks.

vi The tools adopted in the previous investigations can be used not only for synthesizing artificial agents, but also for analyzing scenarios with biological, and indeed human agents. Specifically we consider the problem of the publishing system assessment, where non-observable quantities are explained by some indicators, namely the H-index for the authors, and the Impact factor for the journals. We design a model of the publishing system, in which all the actors (authors and editors) are embodied by artificial agents, and we let the system evolve over time. Through the data collected from this model we estimate the effectiveness of the currently used indicators, showing that H-index is a good proxy for the non-observable authors quality. On the contrary, the Impact factor seems not to be a good predictor of the journals quality in general, but only under specific conditions.

Chapter 1

Background

1.1 Reinforcement Learning

Reinforcement Learning (RL) is a computational approach that allows to learn an agent behavior through the interaction with the environment, by taking into account the consequences of the actions, and by building the connection between actions and rewards. This approach is indeed very rooted in the way animals learn throughout their life, but here adopting the perspective of artificial intelligence research. In this framework there is no teacher, and the agent must discover by itself the behaviour that leads to the highest reward. RL is a term that indicates a problem, a class of solution methods for that problem, and the research field that studies both the problem and the class of solutions as well.

According to the agent-environment interface (Figure 1.1), at time t the agent takes action A_t, given the current environment state S_t, which might not necessarily be completely known by the agent. The action A_t affects the environment at time $t+1$, thus the agent is then rewarded with R_{t+1}, and observes the new environment state S_{t+1}.

At time t the agent objective is to maximize the expected future return over time, namely G_t, as:

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \tag{1.1}$$

Here $\gamma \in [0, 1]$ is called *discount factor*, and it defines how far in the future the agent strategy can reach. If $\gamma = 0$ the agent is *myopic*: it only cares of the immediate reward, and thus its actions will be directed to maximize R_{t+1}, since all the future rewards are worth 0. If $\gamma = 1$ the agent is *farsighted*: it cares of all the future rewards regardless of their position in the future, as if the episode would last forever. The choice of γ therefore determines the optimal behavior the agent is able to find, and is typically set to $0 < \gamma < 1$.

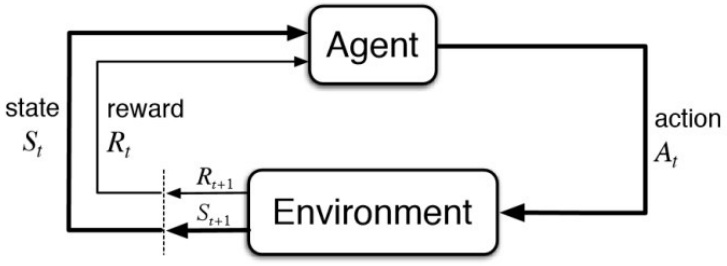


FIGURE 1.1: Agent-environment interface

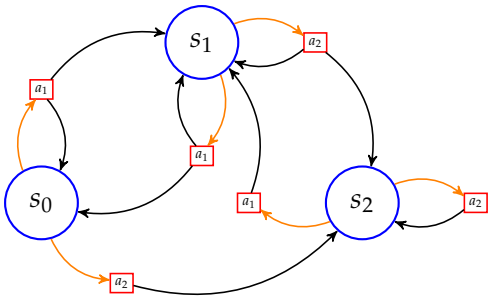


FIGURE 1.2: A MDP example.

The agent-environment interaction in RL is formalized by using the concept of Markov Decision Process (MDP) framework. Given an MDP, it holds the Markov property, that is the state transition function of the process depends only upon the present state, and not on the sequence of past visited states. In a MDP the state transitions are ruled by a transition function $p : S \times R \times S \times A \mapsto [0, 1]$. According to this function p, the probability of transitioning to state s' and receiving reward r, being in state s and taking action a is indicated by $p(s', r|s, a)$. It holds that:

$$\sum_{s' \in S} \sum_{r \in R} p(s', r|s, a) = 1, \forall s \in S, a \in A \tag{1.2}$$

An example of MDP is showed in figure Figure 1.2, where the states S_0, S_1, and S_2 are coloured in blue, and the possible actions from a certain state are denoted by orange arrows. The agent policy therefore associates a probability value for each orange arrow originated from the same state. The possible state transitions given a starting state and the action taken is indicated by a black arrow, therefore the state transition probability function associate a probability value to each arrow with the same origin.

In general the agent policy π is a mapping from each state $s \in S$ to the probability of taking each possible action $a \in A$. If the agent follows policy π at time t, the chance of taking action $a = A_t$ being in state $s = S_t$ is denoted by $\pi(a|s)$. Since π is a probability distribution, it follows that $\sum_{a \in A} \pi(a|s) = 1$.

The *state-value function* for the agent being in a state $S_t = s$ at time t, and following policy π is evaluated as the expectation of the return achieved by following π, that is:

$$V_\pi(s) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} | S_t = s] \tag{1.3}$$

Starting from this formula we can use the definition of expected future return G_t at time t, to present the *Bellman equation* for the state-value function:

$$V_\pi(s) = \sum_{a \in A} \pi(a|s) \sum_{s', r} p(s', r|s, a)[r + \gamma V_\pi(s')] \tag{1.4}$$

Here we indicate $R_{t+1} = r$, $S_{t+1} = s'$, and $A_t = a$ for brevity. The Bellman equation allows to recursively define the relationship between the value of a state S_t and the value of the following state S_{t+1} under a policy π.

It is possible to establish a partial ordering among all the policies, such that the policy π is better than policy π', if the expected return of π is greater or equal than the one of π' for all the possible states. In other words π is better than π' if $V_\pi(s) \geq V_{\pi'}(s), \forall s \in S$. This means that there is an *optimal policy* that we call π^* that

is better than all the others. The optimal policy π^* state-value function is defined for state s by the *Bellman optimality* equation as:

$$V^*(s) = \max_{\pi \in \Pi} V_\pi(s) \tag{1.5}$$

Which can be expanded into:

$$V_{\pi^*}(s) = \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V_{\pi^*}(s')] \tag{1.6}$$

1.1.1 Dynamic Programming

Dynamic Programming (DP) is a collection of algorithms that make it possible to find the agent optimal policy for a task, if the environment model is completely known, and the environment is in the form of a MDP. These algorithms are not always useful since they are computationally expensive, and also because they require perfect knowledge of the environment, i.e., the transition probability function p, which is something that most real life problems lack of.

The *value iteration* algorithm is obtained by turning the Bellman optimality equation into an update rule. If a policy is obtained by blindly following the current state value function, thus exploiting the knowledge on the solution space that are known at that time we call it a greedy policy. This algorithm makes the policy *greedy* w.r.t. the current state-value function, and it is defined for each $s \in S$ as:

$$V_{k+1}(s) = \max_a \sum_{s',r} p(s', r|s, a)[r + \gamma V_k(s')] \tag{1.7}$$

Given an initial guess $V_0(s)$ at time $t = 0$, $\forall s \in S$, this algorithm converges to the optimal state-value function $V^*(s)$.

It is therefore possible to explicitly define the optimal policy π^* as:

$$\pi^*(s) = \arg\max_a \sum_{s',r} p(s', r|s, a)[r + V^*(s')] \tag{1.8}$$

1.1.2 Monte Carlo Methods

Monte Carlo (MC) methods are a collection of algorithms that do not assume complete knowledge of the environment, but instead the update rules of these methods are based on the experiences collected. These methods are based on estimating the expected future returns by means of averaging the returns observed after visiting a certain state. The more returns are observed for each state, the more this estimate converges to the expected value. One downside of these methods is that they do not allow online updates, but the policies are updated only through complete episodes sampling.

Without a model however, estimating the state-value function is not sufficient to find the optimal policy. The *state-action value function* for being in a state $S_t = s$ at time t, and taking action $A_t = a$, and following the optimal policy, is evaluated as:

$$Q^*(s, a) = \sum_{r,s'} p(s', r|s, a)[r + \gamma \max_{a'} Q^*(s', a')] \tag{1.9}$$

This function is estimated using MC methods, by sampling k complete episodes, in which a random policy is employed, and the return for each pair state-action

encountered is averaged through the episodes as:

$$Q_k(s,a) = \mathbb{E}_k[G_t|S_t = s, A_t = a] \tag{1.10}$$

This estimate converges to the true underlying return, for each state-action pair, and given an infinite number of episodes.

It is therefore possible to explicitly compute the optimal policy π^* as:

$$\pi^*(s) = \arg\max_a Q^*(s,a) \tag{1.11}$$

This is an on-policy algorithm, since it improves and evaluates the same policy that is used to sample the episodes state-action trajectory, i.e. that is used to make decisions. Differently the off-policy algorithms optimize one policy, while they sample the episodes state-action trajectory with another.

1.1.3 Temporal Difference Learning

Temporal Difference (TD) approach is a based on the MC idea of directly interacting with the environment rather than learning from a model of the environment, and is based on the DP idea of learning from previous estimates, without needing to complete the episode.

TD methods are based on the TD-error, which is the difference between return estimates at different times into the episode. TD(0) methods update the state-action value function in a one-step fashion, which is based on its value at time k, and the reward received at time $k+1$, updating the estimate with a *learning rate* $\alpha \in [0,1]$. TD(0) algorithms have been proved to converge to the optimal policy, under the condition of a sufficiently small learning rate.

The Q-learning is a TD(0) algorithm that updates the current estimate of the state-action value function being in state $S_k = s$ and taking action $A_k = a$ at time k, through the reward $R_{k+1} = r$ experienced at time $k+1$, regardless of the policy being followed:

$$Q_{k+1}(s,a) = Q_k(s,a) + \alpha\left(r + \gamma \max_{a'} Q_k(s',a') - Q_k(s,a)\right) \tag{1.12}$$

Here the TD-error at time k measures the difference between the return estimate $r + \gamma \max_{a'} Q_k(s',a')$ that is available at time $k+1$, and the less accurate estimate $Q_k(s,a)$ at time k, and is defined as:

$$\delta_t = r + \gamma \max_{a'} Q_k(s',a') - Q_k(s,a) \tag{1.13}$$

1.2 Evolutionary Computation

Evolutionary Computation (EC) is a family of optimization algorithms inspired by biological evolution observed in nature. According to EC, an initial population of candidate solutions compete for limited resources, and therefore is subjected to natural selection (individuals die and are born). Fittest individuals survive and reproduce more than others, and the offspring inherit some traits from their parents, thus gradually moving to regions of the solutions space with higher fitness values. The quality of the solutions w.r.t. a certain problem is determined by the fitness value of that solution, therefore the fitness function drives the evolutionary process. No

assumptions on the fitness landscape, nor on the nature of solution are necessary for solving the problem.

Each individual of the population is internally represented by its *genotype*, which is a convenient representation used by EC. The genotype is subjected to the evolutionary operators (the search for optimal solutions occurs in the genotype space). The same individual is also represented by its *phenotype*, which is the actual candidate solution for the considered problem. The fitness of an individual is computed upon its phenotype. The genotype-phenotype mapping is demanded to the user, and occurs by mapping the genotype space into any arbitrary phenotype space, thus allowing the re-use of EC, regardless of the nature of the problem.

1.2.1 Population Dynamics

In EC individuals life span is instantaneous, and so is the fitness calculation, and therefore the flow of time is determined by the births and deaths.

According to the overlapping generational model, at each time a population of m parents produces a population of n offspring, and the two populations are merged together. From this newly generated population of $m + n$ individuals, only m are selected to survive and become the new population.

On the other side, according to the non-overlapping generational model, at each time a population of m parents produces a population of $n \geq m$ offspring. All the n parents die, and from the n offspring only m are selected to survive and become the new population.

There are many possible selection criteria, but most of the time they are fitness-based criteria. Arguably the most common one is tournament selection, according to which n_{size} individuals are randomly sampled from the population with uniform probability, and among them only survives, that is the one with the higher fitness.

1.2.2 Genetic operators

Reproduction occurs from the initial population of n individuals, where the m offspring genotype is obtained by selecting one or more parents and applying the genetic operators, namely mutation and crossover. The mutation is an unary operator that randomly changes one or more values in the individual genotype. This operator promotes the exploitation of the currently known good solutions, by searching for better solutions close to these in the genotype space.

The crossover is a binary operator that generates an offspring as a recombination of the parents genotype. An offsrping generated using crossover inherits half of its genetic material from each of its parents, but its genotype is very different from both parents. This operator indeed promotes the exploration in the solutions space, by searching for good solutions in very different solutions w.r.t. the currently known ones.

An example of EC pseudocode is presented in Algorithm 1: an initial population is randomly sampled from the solution space using the $Initialize()$ function. An condition based on the number of individuals generated $n_{pop}n_{gen}$ forces the algorithm to exit the main loop, where parents are selected using $SelectTournament()$. The

selected parents are then variated using the genetic operators o_m, and o_c, and these new individuals are used to produce the new population.

```
1  b ← 0
2  I = Initialize()
3  while b ≤ n_pop n_gen do
4      I′ = ∅;
5      for i ∈ 1, ..., n_pop do
6          (g_p1, p_p1, f_p1) ← SelectTournament(l)
7          (g_p2, p_p2, f_p2) ← SelectTournament(l)
8          g_c ← o_m(o_c(g_p1, g_p2))
9          I′ ← I′ ∪(g_c, φ(g_c), f(φ(g_c)))
10         b ← b + 1
11     end
12     I ← I ∪ I′
13     while |l| > n_pop do
14         I ← I
15         SelWorst(l)
16     end
17 end
```

Algorithm 1: An EC pseudocode example

1.3 Multi-agent System

A multi-agent system (MAS) is a mathematical abstraction of a certain aspect of reality, in which multiple intelligent agents interact. Multi-agent might be an inherent property belonging to the model considered, but might also be a convenient representation for dividing a complex problem into its simple constitutive parts. A multi-agent system might contain passive objects and active agents. The passive objects do not interact in any way with the environment, and represent simply obstacles, resources, or goals for the agents, while the active agents are actually the only entities capable of interacting with the passive objects of the environment, and with other agents to some extent, and are typically subjected to a learning process. Let us consider the MAS example of the predator-prey model. In this MAS, the active agents, namely predators and preys, are designed with a certain degree of autonomy, i.e. they are free to act on its own, without no need for an external control. The agents within the environment might be defined by different features and different goals, i.e. preys and predators are by design different kinds of agents, and have different goals: the preys aim at their own survival, while the predators aim at catching them. Moreover the formalization of the MAS might not allow the agents to have a complete knowledge of the system internal state, i.e. the predators might not be able to sense the preys if they are not close enough. Another relevant aspect for MAS is the decentralization, according to which each agent is responsible for the scope of its learning, and there is no central authority responsible for supervising each agent. Finally MAS can manifest self-organization and self-direction, which allow to promote complex collective behaviors even when the individual strategies are simple. This complexity aspect is enforced in particular when the agents are provided with a way to share their knowledge, as it occurs when employing communication: the predators might emit sounds to establish a hunting strategy, and conversely the preys might communicate the presence of a predator to alert the others.

1.4 Research Questions

In this work we consider relevant and partially unexplored aspects of multi-agent scenarios, such as the optimal communication setting for improving both efficiency and effectiveness of RL, or such as the impact of communication on the trade-off individual-collective behavior that is developed by RL in different scenarios, or finally such as the regulation of AIs by teaching them, through a framework of flexible rules based on RL, to avoid the disallowed behaviors. We then investigate the possibility of automatically design and control the next generation of nature-inspired soft robots through EC, by proposing a sensing control strategy which might take advantage of the body complexity, and we compare its performance with the one of a controller from the literature, and finally by suggesting an approach to automatically design adaptable soft robot bodies that are successful in tasks requiring different skills. Finally we adopt the multi-agent framework for assessing the effectiveness of the currently used indicators for assessing the quality of the publishing system actors. We formalize here each one of these aspects in the form of research questions, and we briefly introduce the content of each chapter that is relevant to each one of these questions.

1.4.1 Research Question 1

Which is the optimal language expressiveness for learning the most effective and efficient behaviors in a cooperative multi-agent RL?

This question is addressed in Chapter 2, in which for a given a cooperative multi-agent communication game in which each agent strategy is learned through RL, the language expressiveness is function of the vocabulary size and of the problem complexity as well. Different scenarios and communication settings for the same game allow to train the agents under different condition, which are indeed responsible for defining the training complexity. Each value of language expressiveness considered is evaluated in terms of both learning efficiency, i.e. how many learning iterations are necessary for the agents to develop the optimal strategy, and effectiveness, i.e. how good is the agents learned strategy.

1.4.2 Research Question 2

What is the impact on the individual-collective reward relationship of learning the communication strategies within a Multi-Agent RL game?

In Chapter 3 we consider three different scenarios for a multi-agent resources collection game, namely a cooperative, a competitive, and a mixed one. Each scenario is designed in a way to drive the agents learning towards a different optimal behavior w.r.t. the individual and collective goal. In all these scenarios presented, the interaction among agents is possible only in the form of a hard-coded broadcast communication. Every agent independently learns an optimal behavior for each presented game, where the individual policy includes the possibility to enable or disable the broadcast communication at any step of the game. We then investigate the impact of the individually learned communication strategies on the overall collective behavior, by measuring the individual return of the learned policies and the inequality in the resources allocation.

1.4.3 Research Question 3

Is it possible to provide regulations for autonomous vehicles without explicitly disabling unethical behaviors?

To answer this question, we consider a simulated version of a road traffic scenario, in which each driver interacts with its vehicle by means of updating the speed and by changing lanes, and where each vehicle is controlled by a RL agent, as described in Chapter 4. A form of regulation inspired by the real traffic rules is enforced as negative reward received by the agents when evading any of the rules, while being trained towards traffic efficiency, i.e. high average speed, and safety, i.e. no collisions. In Chapter 4 we stress the importance of this flexible AI regulation approach, which might be relevant to the ethical problem of autonomous vehicles. We thus compare the results of this regulation approach with another variant of the training, in which the rules enforcement is missing, and we measure the impact of the rules on the validation results.

1.4.4 Research Question 4

Is it possible to take advantage of the body complexity by evolving a sensing neural controller for voxel-based soft robots?

One of the underlying ideas of the voxel-based soft robot technology is the shift of complexity from the brain to the body of these robots. Due to this complexity, finding the optimal control algorithm for such robots is a challenging task that has been automatically solved in the literature through EC. The form of control proposed in other works are often based on non-sensing time signals, which are responsible for actuating each voxel accordingly. In Chapter 6 we propose to leverage on the body complexity by providing sensors placed in each voxel of the robot body, and a control algorithm in the form of an artificial neural network. We then estimate the capabilities of the sensing approach, by comparing the performance of the two control variants on a set of manually designed bodies, and different environments for a locomotion task.

1.4.5 Research Question 5

Is it possible to automatically design a body for voxel-based soft robots that can adapt to differnt tasks?

In Chapter 6 we consider adaptability as a property belonging to a voxel-based robot body, and defined as the ability to perform well at tasks that require different skills. A body that lacks this property might possess only the skills necessary to accomplish a subset of these tasks, while performing poorly on the others. In this chapter we propose a way to measure adaptability based on the self-organized criticality, which is a property of systems which allows the most complex dynamics. A number of bodies are optimized by means of EC, using the definition of self-organized criticality as a proxy for the adaptability. We design several other bodies, using the results from Chapter 5 as well as others from the literature, together with other baselines, and we experimentally compare all of them on different tasks, in order to validate this optimization approach for designing adaptable bodies.

1.4.6 Research Question 6

Are the currently employed publishing system indicators effective proxies for the corresponding non-observable quantities they embody?

To answer this research question, in Chapter 8 we design a simulation of the publishing system, where the authors and the editors are modeled by artificial agents. In this simulation the authors write and submit their papers, while the editors either accept or reject the papers submitted to their journal. Agents are provided with some manually designed policies, and with others learned by means of RL as well. Given each combination of policies employed by the agents, we let the system evolve over time, and we collect the resulting data from the simulation. From the data collected using this model, we empirically measure the effectiveness of the currently employed indicators for assessing respectively the authors through the H-index, and the journals through the Impact factor.

Chapter 2

How the Language Expressiveness affects Reinforcement Learning

A *cooperative multi-agent* system is a category of multi-agent systems in which agents individual and collective goals are the same. In this chapter we consider a cooperative multi-agent system, in which cooperation may be enforced by communication between agents but in which agents must *learn to communicate*. The system consists of a game in which agents may move in a 2D world and are given the task of reaching specified targets. Each agent knows the target of another agent but not its own, thus the only way to solve the task is for the agents to guide one another using communication and, in particular, by learning how to communicate. We cast this game in terms of a partially observed Markov game and show that agents may learn policies for moving and communicating in the form of a neural network by means of *reinforcement learning*. We investigate in depth the impact on the learning quality of the *expressiveness* of the language, which is a function of vocabulary size, number of agents and number of targets.

2.1 Introduction

Artificial intelligence nowadays plays an increasingly pervasive role in everyday life and what was only a research topic a few years ago is now an essential part of the technology industry (Syam and Sharma, 2018). An important goal in this scenario consists in the creation of artificial agents able to solve complex problems requiring forms of perception of the surrounding environment. Multi-agent systems, in particular, involve many artificial agents interacting in the same environment with common and/or conflicting goals, resulting in cooperative and/or competitive scenarios.

For example, the controller of an autonomous vehicle can be modeled as a multi-agent system of sensors and actuators that work together, so that collective performance depends on the cooperation among all of them. The cooperation involves the exchange of information between agents along with the evaluation of the action to take, based on the received information.

Language interpretability and efficiency are important requirements in this context. Natural language has the advantage of being understandable by humans, but its complexity and the fact that the meaning of a term may depend on the context in which it appears (i.e., on nearby terms) may nullify the potential benefits of the learning process (Szabó, 2017).

A symbol-based language, with no predefined meaning for symbols, is not easily interpretable by humans, but could lead to more efficient cooperation among artificial agents and could be more appropriate for a learning task than natural language.

Motivated by recent works on multi-agent systems (Lowe et al., 2017; Mordatch and Abbeel, 2017), we consider a *symbol-based* approach to solve a cooperative multi-agent task in which agents learn to communicate using *reinforcement learning* (RL). We consider a multi-agent scenario in which agents may move in a 2D world and are given the task of reaching specified targets. Each agent knows the target of another agent but not its own, thus the only way to solve the task is for the agents to guide one another using communication and, in particular, by learning how to communicate.

We use the Multi-Agent Deep Deterministic Policy Gradient (MADDPG) learning algorithm (Lowe et al., 2017) for learning the control policy of the agents in a fully decentralized way. We focus on investigating the relation between vocabulary size (number of communication symbols available), number of agents, and number of possible targets. To this end, we define a single *expressiveness* numerical index that depends on these quantities and investigate its effect on the degree of accomplishment of the cooperative task at the end of the learning process.

Given an instance of the problem, with a fixed number of agents and targets, we find a vocabulary size that leads to best rewards w.r.t. the number of agents and targets. We compare these results to two baselines that model the two extreme scenarios: one in which agents cannot communicate and another in which agents have a predefined ability to communicate. Finally we show that, if expressiveness is beyond a certain threshold, the learning process struggles to learn how to communicate and, consequently, performs poorly.

2.2 Related work

Our work concerns the language used for communication among agents in a cooperative scenario where the learning of agent controllers is tackled with RL. In the following sections, we briefly survey relevant previous studies related this scenario.

In particular, we focus on language and communication in multi-agent systems (Section 2.2.1) and on RL as tool for training the agent controllers in these settings (Section 2.2.2). Indeed, most of the cited works involve both aspects. We decide to present each work in the section related to the work most relevant part of the contribution.

It is worth to note that our work somewhat investigates the broader field studying how the (natural) language has evolved. The reader may find in Christiansen and Kirby, 2003 a comprehensive analysis of that field.

2.2.1 Language and communication

In many works the agents are viewed as a complex adaptive system which collectively solves the problem of developing a shared communication model (Steels, 2000a). To do so, the community must reach an agreement on a repertoire of forms, of meanings, and of form-meaning pairs (the lexicon and grammar). In general, this task may be hard: the authors of Kirby, Griffiths, and Smith, 2014 discuss the learning bottleneck and its relation with the phrase structure and, more broadly, with language compositionality.

In the communication-oriented multi-agent scenario of Havrylov and Titov, 2017, the authors implement the agents through recurrent neural networks, allowing language compositionality and variability. Their focus is on the development of natural language, and the results are obtained by fixing the vocabulary size to an arbitrary value. The works cited above are based on language compositionality and grammar complexity, an aspect we do not deal with.

In Yamada et al., 2017 the authors train a robotic agent controlled by a recurrent neural network to learn the relationship between sequences of words and corresponding actions. They consider a symbol-based language in which every word is encoded as one-hot vector and visualize the agent internal representation of the state after the training. In the cited work there is no multi-agent interaction, as in our work, but a single agent model is trained to learn sentence-action associations. Moreover, rather than focusing on expressiveness, Yamada et al., 2017 assesses the quality of the learning w.r.t. the presence of specific logical expression in the uttered word sequences.

In Bachwerk and Vogel, 2011 the authors claim that language is a mechanism that emerges for coordinating the solution of complex tasks among agents. In Bachwerk and Vogel, 2012 the same authors show that different level of "friendship" between agents co-evolve with a system of linguistic conventions: agents decisions during individual interactions influence the overall social structure of the population. Both works consider a tool, called *Language Evolution Workbench* (LEW), which defines the task, the scenario, and the learning procedure. Based on simulations using LEW, they study the impact of the probability of communication success on the task achievement: moreover, they characterize the former in terms of lexicon size and amount of lexicon actually used. This analysis is similar to the one we do, even if the scenario and the task are different.

2.2.2 Reinforcement learning for agent controllers

RL has reached outstanding results in single-agent control domain (Duan et al., 2016) and several recent works have extended this approach to the training of cooperative multi-agent systems, also when communication among agents plays a role (Steels, 2000a). In Steels, 2000a and in Havrylov and Titov, 2017 communication is enforced

in a referential game between two cooperating agents: one agent goal is to explain which image the other agent should select from a pool of images.

In Foerster et al., 2016 the authors investigate the process of learning to communicate and how this occurs among agents. To this extent they propose two multi-agent frameworks for communicative tasks based on reinforcement learning.

The focus of Steels, 2000b is on the meaning-word association in a population of autonomous real robots. The robots play a guessing game in which one has to communicate the other that he identified an object, acquired through a camera. The works cited above use RL to learn the agent controllers, including the communication part, but focus on different scenarios: in particular, they deal with cases where the communication involves two agents.

Recently, another RL variant tailored to communication-based tasks has been presented in Bahdanau et al., 2018. The authors propose to approximate the reward function, used to assess the agent behaviour, with a discriminator neural network trained on a data set of states. In this way, the representation of the solved task is separated from how to solve it. This approach has been showed to outperform vanilla RL on a range of tasks, and combines RL with supervised learning.

Another fully decentralized multi-agent RL approach for commu-nication-based tasks has been proposed by Vrieze et al., 2018. The cited work shows that agents may indeed learn to communicate without any prior information. Agents learn a low-level wireless protocol with bidirectional communication. As in our work, the authors also provide baselines for both transmitter and receiver, and they learn only one of the two. Training quality is assessed with varying levels of noise in the communication channel.

The approach that we use for training the agents is Multi-Agent Deep Deterministic Policy Gradient (MADDPG) (Lowe et al., 2017), a multi-agent variation of deterministic policy gradient (Silver et al., 2014). The cited work shown that MADDPG fits several different RL scenarios, and proposed its application in many cooperative and competitive multi-agent settings.

In Mordatch and Abbeel, 2017 and Havrylov and Titov, 2017 the authors present multi-agent scenarios with communication, in which an end-to-end differentiable model is trained through backpropagation. In particular in Havrylov and Titov, 2017 this type of learning is compared with standard RL.

We consider a scenario similar to the one in Mordatch and Abbeel, 2017, in which agents move in a 2D word and communicate using symbol-based language and agent controllers are implemented by neural networks. In the cited work, the authors consider an end-to-end differentiable approach for training: they assume differentiable system dynamics, so that the networks can be updated back-propagating an arbitrary reward scalar. This work is different from our work in terms of learning approach. Also the cited work does not evaluate the impact of expressiveness on the training results, focusing instead on analyzing the actual number of words emitted by the agents with varying vocabulary size.

A mixed cooperative-competitive scenario has been studied by Lewis et al., 2017, where two opposing agents have to negotiate a deal, in order to maximize their opposite goals. The two agents use natural language for the communication and the baseline for the training is supervised learning using a data set of dialogues on negotiation tasks. The agents are then trained from this baseline by using RL and self-play. Again we do not use supervised learning, nor natural language, and the agents in our work play a cooperative game, instead of competing against each other.

2.3 Background

2.3.1 Markov game

We consider a multi-agent version of a Markov Decision Process (MDP), called *partially observed Markov game* Littman, 1994 (or, here briefly, Markov game), in which a number of *agents* interact in the same discrete-time environment.

A Markov game involving n agents is described by a tuple $(S, \phi, \rho, \mathcal{O}, \mathcal{A}, \Omega, \Pi, R)$, where S is the set of possible *states* of the game, ϕ is the stochastic *state transition function*, ρ is the stochastic *initial state function*, $\mathcal{O} = (O_1, \ldots, O_n)$ is the set of agent-dependent *observations sets*, $\mathcal{A} = (A_1, \ldots, A_n)$ is the set of agent-dependent *actions sets*, $\Omega = (\omega_1, \ldots, \omega_n)$ is the set of agent-dependent stochastic *observation functions*, $\Pi = (\pi_1, \ldots, \pi_n)$ is the set of agent-dependent stochastic *policies*, and $R = (r_1, \ldots, r_n)$ is the set of agent-dependent *reward functions*.

The state transition function $\phi : S \times S \times A_1 \times \cdots \times A_n \mapsto [0, 1]$ stochastically determines the evolution of the game, i.e., $\phi(s'|s, a_1, \ldots, a_n)$ is the probability that the game goes from state s to state s' given that the agents performed the actions $a_1, \ldots, a_n$. It holds that $\forall s \in S, \forall (a_1, \ldots, a_n) \in A_1 \times \cdots \times A_n : \sum_{s' \in S} \phi(s'|s, a_1, \ldots, a_n) = 1$.

The initial state function $\rho : S \mapsto [0, 1]$ stochastically determines the initial state of the game, i.e., $\rho(s)$ is the probability that the game starts from the state s. It holds that $\sum_{s \in S} \rho(s) = 1$.

Each observation function $\omega_i : O_i \times S \times A_i \mapsto [0, 1]$ stochastically determines the observation of the corresponding agent, i.e., $\omega_i(o|s, a)$ is the probability that the ith agent observes the observation $o \in O_i$, given that it performed the action $a \in A_i$ with the game being in state s. It holds that $\forall i \in \{1, \ldots, n\}, \forall s \in S, \forall a \in A_i : \sum_{o \in O_i} \omega_i(o|s, a) = 1$.

Each policy $\pi_i : O_i \times A_i \mapsto [0, 1]$ stochastically determines the behavior of the corresponding agent, i.e., $\pi_i(a|o)$ is the probability that the ith agent performs the action $a \in A_i$ given that it observed the observation $o \in O_i$. It holds that $\forall i \in \{1, \ldots, n\}, \forall o \in O_i : \sum_{a \in A_i} \pi_i(a|o) = 1$.

Each reward function $r_i : S \times A_i \mapsto \mathbb{R}$ determines the reward an agent receives, i.e., $r_i(s, a)$ is the reward the ith agent receives for having performed the action $a \in A_i$ with the game in state $s \in S$.

2.3.2 Policy learning

Let $\gamma \in [0, 1]$ be a predefined constant called *discount factor* and let T be a predefined time length called the *time horizon*. The *policy learning problem* consists in finding, given $S, \mathcal{O}, \mathcal{A}, R$ and γ, T, the *optimal policies* $\Pi^\star$ which maximize the expectation $\mathbb{E}[\bar{r}^{t_0}]$, of the *overall reward* $\bar{r}^{t_0}$ for any t_0, defined as

$$\bar{r}^{t_0} = \sum_{i \in \{1, \ldots, n\}} \sum_{t=t_0}^{t=t_0+T} \gamma^t r_i(s^t, a_i^t) \tag{2.1}$$

where s^t is the state of the game at the tth step and a_i^t is the action performed by the ith agent at the tth step. Parameters γ and T specify to which degree the agents employing the optimal policies $\Pi^\star$ act towards an immediate reward (short T, small γ) or a future reward (long T, $\gamma \approx 1$).

It can be noted that the state transition function ϕ, the initial state function ρ, and the observation functions Ω are not available in the policy learning problem. Instead, it is possible to sample the corresponding distribution by playing the game as many

times as needed. That is, given a policies set Π, it is possible to play the game for a finite number T_{episode} of time steps (i.e., an *episode*) and obtaining the corresponding values of s^t, o_i^t, a_i^t for $t \in \{0, \ldots, T_{\text{episode}}\}$—and, as a consequence, the rewards $r_i(s^t, a_i^t)$ for $t \in \{0, \ldots, T_{\text{episode}}\}$ and the overall rewards $\bar{r}^t$ for $t \in \{0, \ldots, T_{\text{episode}} - T\}$.

In many cases of interest, the policies to be learned have all the same form (e.g., a neural network) which can be modeled by a finite set θ of numerical *policy parameters*. In those cases, the policy learning problem consists in learning the optimal values $\Theta^\star = \{\theta_1^\star, \ldots, \theta_n^\star\}$ of the policy parameters.

The policy learning problem can be solved using RL, a form of machine learning which tries to balance exploration (i.e., sampling ϕ, ρ, and Ω) and exploitation (i.e., maximizing the overall rewards $\bar{r}$) while searching in the space of the policy parameters by varying Θ. In this work, we use a RL technique called Multi-Agent Deep Deterministic Policy Gradient (MADDPG) Lowe et al., 2017 which has been shown to be effective for scenarios similar to the one considered in this work.

2.4 The Cooperative Unknown Target game

We consider a cooperative game similar to the one presented in Mordatch and Abbeel, 2017, which we call the *Cooperative Unknown Target* (CUT) game. The game is based on 2D world in which n_r robots move and communicate. The goal of each robot is to reach one among n_t target positions in the world, the association between robot and target being statically determined before the game starts. Each robot knows the target of one robot but *not* its own target. It is guaranteed that the target of each robot is known to exactly one robot. Robots communicate by sending and receiving symbols of a finite alphabet W (the *language*). At each time step, a robot broadcasts exactly one symbol: each sent symbol is received by every robot and the sender is known to the receivers. Robots are stateless: in particular, they do not remember the symbols they received before the current time step.

Communication among robots is thus essential for allowing each robot to reach its own target, which characterizes the CUT game as a cooperative one. The game is challenging because a robot should learn to: (i) move toward its target, to be deduced from what it hears; (ii) broadcast a symbol describing the single robot-target association that it knows.

The CUT game can be modeled as a Markov game. In the following sections, we detail how the CUT game maps to the Markov game abstraction and which is the form we chose for the corresponding policies, that we then learned using MADDPG. As anticipated in Section 5.1, the aim of the present paper is to investigate the impact of the language size on the effectiveness and efficiency of the policy learning; we do not insist on discussing the optimality of the game mapping, nor on thoroughly evaluating the MADDPG performance on the CUT game.

2.4.1 The CUT game as a Markov game

The key idea behind the mapping of the CUT game to the Markov game abstraction is to represent each robot in the former as two separate agents in the latter: one (the *movement-agent*) determines the movements of the robot, the other (the *communication-agent*) determines the output communication of (i.e., the symbols sent by) the robot. The motivation for this choice is to allow for the decoupling of the two different goals of the robot: (i) moving towards its target and (ii) communicating the known

robot-target association. In particular, the two corresponding reward functions can be hence defined to reflect the two goals.

More in detail, the state of the CUT game encodes:

- the positions $\boldsymbol{x}_{r,1}, \ldots, \boldsymbol{x}_{r,n_r}$ of the n_r robots, with $\boldsymbol{x}_{r,i} \in \mathbb{R}^2$ for each i;
- the speeds $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{n_r}$ of the n_r robots, with $\boldsymbol{v}_i \in \mathbb{R}^2$ for each i;
- the positions $\boldsymbol{x}_{t,1}, \ldots, \boldsymbol{x}_{t,n_t}$ of the n_t targets, with $\boldsymbol{x}_{t,i} \in [0,1]^2$ for each i;
- the robot-target associations $(\tau_{r,1}, \tau_{t,1}), \ldots, (\tau_{r,n_r}, \tau_{t,n_t})$ known by each robot, with $\tau_{r,i} \in \{1, \ldots, n_r\}$ and $\tau_{t,i} \in \{1, \ldots, n_t\}$ for each i;
- the symbols $w_1, \ldots, w_{n_r}$ sent by n_r robots at the previous time step, with $w_i \in W \cup \varnothing$ for each i.

All the robots in the CUT game can move and communicate in the same way. Let $A^{\text{move}} = \{\uparrow, \rightarrow, \downarrow, \leftarrow, \varnothing\}$ be the set containing the actions which can be performed by the movement-agent of a robot. Values of A^{move} represent the changes of speed in the direction, while $\varnothing$ representing the null change. Let $A^{\text{comm}} = W \cup \varnothing$ be the set containing the symbols which can be sent by the communication-agent of a robot, $\varnothing$ representing no sent symbol.

The mapping between the CUT game and the tuple $(S, \phi, \rho, \mathcal{O}, \mathcal{A}, \Omega, \Pi, R)$ defining a Markov game is as follows. The set $\mathcal{A}$ of action sets of the Markov game is composed of n_r copies of A^{move} and n_r copies of A^{comm}. Similarly, the set $\mathcal{O}$ of observation sets of the Markov game is composed of n_r copies of the set $O^{\text{move}} = \mathbb{R}^2 \times \cdots \times \mathbb{R}^2 \times W \cup \varnothing \times \cdots \times W \cup \varnothing$, where $\mathbb{R}^2$ is repeated n_t times and $W \cup \varnothing$ is repeated n_r times, and n_r copies of $O^{\text{comm}} = \{1, \ldots, n_r\} \times \{1, \ldots, n_t\}$. The semantics of the two sets $O^{\text{move}}, O^{\text{comm}}$ is explained below in terms of the observation functions Ω.

The initial state function ρ of the Markov game sets the state variables $\boldsymbol{x}_{r,i}, \boldsymbol{x}_{t,i}$ randomly in $[0,1]^2$, $(\tau_{r,i}, \tau_{t,i})$ randomly in the corresponding domains, $\boldsymbol{v}_i = (0,0)$, and $w_i = \varnothing$, for each i (we do not formalize this function for the sake of brevity). Concerning the robot-target associations $(\tau_{r,i}, \tau_{t,i})$, ρ ensures that (i) no robots know their robot-association and (ii) each robot-target association is known by at least one robot, i.e., $\forall i, \tau_{r,i} \neq i$ and $\forall i, \exists! j \neq i : \tau_{r,j} = i$.

The state transition function ϕ of the Markov game is not actually stochastic (we do not formalize this function for the sake of brevity). It updates the state variables corresponding to the robot positions $\boldsymbol{x}_{r,i}$ according to the actions performed by the corresponding movement-agents and updates the state variables w_i according to the symbols sent by the communication-agents. The other state variables are never affected, i.e., targets do not move and robot-target associations do not change. Concerning robot positions, $\boldsymbol{x}_{r,i}$ becomes $\boldsymbol{x}_{r,i} + \boldsymbol{v}_i$ and $\boldsymbol{v}_i$ becomes $\boldsymbol{v}_i + \Delta\boldsymbol{v}$ with $\Delta\boldsymbol{v}$ being $(-\delta, 0)$, $(0, \delta)$, $(\delta, 0)$, $(0, -\delta)$, $(0,0)$ for, respectively, $\uparrow$, $\rightarrow$, $\downarrow$, $\leftarrow$, $\varnothing$. Parameter δ represents the robots acceleration.

The observation functions of the Markov game are not actually stochastic and depend only on the state (not on the performed action). We denote the state with s. For movement-agents, $\omega_i^{\text{move}} : S \mapsto O^{\text{move}}$ gives the offset of the ith agent from each target along the two axes (a pair of values in $\mathbb{R}$ for each target) and the symbols emitted at the previous step by each agent (one symbol in $W \cup \varnothing$):

$$\omega_i^{\text{move}}(s) = (\boldsymbol{x}_{t,1} - \boldsymbol{x}_{r,i}, \ldots, \boldsymbol{x}_{t,n_t} - \boldsymbol{x}_{r,i}, w_1, \ldots, w_{n_r}) \tag{2.2}$$

For communication-agents, $\omega_i^{\text{comm}} : S \mapsto O^{\text{comm}}$ gives the robot-target association known to the ith robot:

$$\omega_i^{\text{comm}}(s) = (\tau_{r,i}, \tau_{t,i}) \tag{2.3}$$

The robot-target associations $\tau_{r,i}, \tau_{t,i}$ never change, thus every communication-agent observes a constant observation during the game.

Finally, the reward functions of the Markov game are defined so as to capture the goal of the CUT game. For the movement-agents, r_i^{move} rewards the agent when it is able to get closer to its target. For communication-agents, r_i^{comm} rewards the $i$agent when it is able to make its "recipient" jth to get closer to its target, with $j = \tau_{r,i}$. In detail, $r_i^{\text{move}} : S \mapsto \mathbb{R}$ gives the opposite of the Manhattan distance (i.e., the closer, the greater the reward) of the corresponding ith robot from its target:

$$r_i^{\text{move}}(s) = \sum_{j=1}^{j=n_r} -d_1(\boldsymbol{x}_{r,i}, \boldsymbol{x}_{t,\tau_{t,j}}) \mathbb{1}_i(s,j) \tag{2.4}$$

where:

$$\mathbb{1}_i(s,j) = \begin{cases} 1 & \text{if } \tau_{r,j} = i \\ 0 & \text{otherwise} \end{cases} \tag{2.5}$$

The reward function $r_i^{\text{comm}} : S \mapsto \mathbb{R}$ gives the opposite of the Manhattan distance of the robot whose target association is known to the ith robot from its target:

$$r_i^{\text{comm}}(s) = -d_1(\boldsymbol{x}_{r,\tau_{r,i}}, \boldsymbol{x}_{t,\tau_{t,i}}) \tag{2.6}$$

Both reward functions depend only on the state (not on the performed action) and are deterministic.

2.4.2 Policies form

We consider policies in the form of a neural network, thus the policy learning problem consists in learning the optimal values $\Theta^\star = \{\theta_1^\star, \ldots, \theta_n^\star\}$ of the parameters of the network. We opted for a Multi-Layer Perceptron (MLP) as the form of the policies for the movement- and communication-agents, which can hence be denoted as $\pi_{\theta_i}^{\text{move}} : O^{\text{move}} \mapsto A^{\text{move}}$ and $\pi_{\theta_i}^{\text{comm}} : O^{\text{comm}} \mapsto A^{\text{comm}}$, respectively. We chose to use the same MLP topology for all the movement-agents policies and the same MLP topology for all the communication-agents policies.

The input of $\pi_{\theta_i}^{\text{move}}$ consists of a tuple of $2n_t + n_r$ elements, as defined in Section 2.4.1. The first $2n_t$ elements are the offset of the ith agent from each target along the two axes; each of these elements is mapped to the input layer directly. The remaining n_r elements are the symbols emitted at the previous step by each agent (i.e., one symbol in $W \cup \varnothing$); each of these elements is represented in one-hot encoding, thus each of these elements correspond to $|W| + 1$ input values. The resulting size for the input layer will be $2n_t + n_r(|W| + 1)$. The output of $\pi_{\theta_i}^{\text{move}}$ is an element of A^{move} (Section 2.4.1). This element is represented in one-hot encoding, thereby resulting in $|A^{\text{move}}| = 5$ output neurons.

The input of $\pi_{\theta_i}^{\text{comm}}$ consists of a pair of elements corresponding to the robot-target association known to the ith agent $\tau_{r,i}, \tau_{t,i}$, as defined in Section 2.4.1. Both elements are represented in one-hot encoding, resulting in $n_r + n_t$ input values. The output of $\pi_{\theta_i}^{\text{comm}}$ is an element of $A^{\text{comm}} = W \cup \varnothing$ (Section 2.4.1). This element is mapped to $|W| + 1$ output neurons with one-hot encoding.

For both the policies, we set 2 fully connected hidden layers, each with 64 neurons, and we used the Rectifier Linear Unit (ReLU) activation function as done in Lowe et al., 2017. Finally, the policies output are sampled from the Gumbel-Softmax distribution Jang, Gu, and Poole, 2016 which makes the policies actually stochastic: this, in turn, allows for a better exploration of the search space while learning the policies with MADDPG.

2.4.3 Hand-made communication-agents policies

In order to allow for a more insightful investigation of the impact of the language size on the policy learning effectiveness and efficiency, we designed two communication-agents policies to use as baselines. The two policies represent two extreme cases, one of non-communicating agents and one of agents which optimally communicate with a language of predefined size.

The first hand-made policy, which we denote with *NoComm*, is defined by:

$$\pi^{\text{comm}}_{\text{NoComm}}(\tau_{r,i}, \tau_{t,i}) = \varnothing \tag{2.7}$$

Note that, since the output of $\pi^{\text{comm}}_{\text{NoComm}}$ is always $\varnothing$, this policy works with any W: in particular, it works with $W = \emptyset$.

The second hand-made policy, which we denote with *Opt*, is defined by:

$$\pi^{\text{comm}}_{\text{Opt}}(\tau_{r,i}, \tau_{t,i}) = w^{\text{Opt}}_{\tau_{r,i},\tau_{t,i}} \tag{2.8}$$

where $w^{\text{Opt}}_{\tau_{r,i},\tau_{t,i}}$ is a symbol of a language W^{Opt} that can express all the possible robot-agent associations. That is, in W^{Opt} each robot-agent association is unequivocally encoded by one symbol: $W^{\text{Opt}} = \{w_{i,j}, i \in \{1, \ldots, n_r\}, j \in \{1, \ldots, n_t\}\}$.

2.5 Experimental evaluation

2.5.1 Procedure

We emphasize that each agent learns to communicate and to understand the received symbols independently of the other agents. Thus, a given symbol could be associated with different meanings by the communication-agent policy $\pi^{\text{comm}}_{\theta_i}$ of different agents. It is thus important to gain insights into the relation between the number of symbols available for communication $|W|$, the number of robots n_r, and the number of targets n_t.

In order to decouple the experimental findings from the size of the policy learning problem (i.e., the number n_r of robots and the number n_t of targets), we define the *expressiveness* of the language as $e = \frac{|W|}{n_r n_t}$. Others may define expressiveness in different ways, for instance as the number of times symbols have been uttered. We investigated the impact of expressiveness on the e and efficiency of the policy learning in the CUT problem.

We performed policy learning of both π^{move} and π^{comm} for a number of different combinations of $(n_r, n_t, |W|)$ corresponding to $e \in [0, 4]$, as follows. We considered all combinations of $(n_r, n_t) \in \{2, 3, 4\} \times \{2, 3, 4\}$; for each such combination we considered all values for $|W| \in \{0, \ldots, 4 n_r n_t\}$.

For each combination of $(n_r, n_t, |W|)$, we executed an experiment consisting of 20 000 learning episodes followed by the evaluation of the resulting policies for 100 validation episodes. The learning parameters are listed in table 6.1.

TABLE 2.1: Parameters.

	Parameter	Value
Pol. l.	Discount factor γ	0.95
	Time horizon T	25
	Episode time steps T_{episode}	25
CUT	Number of robots n_r	$\in \{2, 3, 4\}$
	Number of targets n_t	$\in \{2, 3, 4\}$
	Language size $\lvert W \rvert$	$\in \{0, \dots, 4n_r n_t\}$
MADDPG	Batch size	512
	Replay memory size	10^6
	Optimizer	Adam
	Learning rate	10^{-2}
	Gradient norm clipping	0.5

For the baseline policies we performed policy learning only for the agent movement policy $\pi^{\text{move}}_{\theta_i}$, because agent communication policies $\pi^{\text{comm}}_{\text{NoComm}}$ and $\pi^{\text{comm}}_{\text{Opt}}$ were specified in advance. We considered all combinations of $(n_r, n_t) \in \{2, 3, 4\} \times \{2, 3, 4\}$, each combination with a single value for $|W|$, as follows. With $\pi^{\text{comm}}_{\text{NoComm}}$ the language is $W = \emptyset$, thus $|W| = e = 0$. With $\pi^{\text{comm}}_{\text{Opt}}$ the language size is $|W| = n_r n_t$, thus the expressiveness is $e = 1$.

2.5.2 Results and discussion: effectiveness

In this section we assess the effectiveness of the policy learning in all the problem settings of the experimental campaign. To this end, we computed the *validation reward* R_{val} for each experiment, defined as the average overall reward of the agents on the validation episodes. Figure 2.1 shows the normalized validation reward averaged across all the experiments—i.e., for each (n_r, n_t) the values of R_{val} for different e are adjusted to have null mean and standard deviation equal to 1.

The most important finding is that the highest validation reward corresponds to values of e close to 1, i.e., when $|W|$ is close to $n_r n_t$. On the other hand, when $e \gg 1$ the action space grows unnecessarily and the learning process is negatively affected. In principle, one could try to tackle the growth of the action space by increasing the size of the MLP: we did not perform any experiments to investigate this opportunity.

Figure 2.1 also shows that the validation reward with $e = 0$ is poor. This is not surprising, because when the language does not allow to communicate, an agent does not have any information about its target, thus the learned policy can only tend to minimize the distance between a robot and all targets. In summary, this figure shows that the policy learning process is indeed effective in learning how to communicate, in order to solve the specific cooperative task. Furthermore, it shows that the policy learning is most effective when the size of the language is sufficiently large to (potentially) allow associating approximately one symbol with each possible robot-target association.

Figure 2.2 provides the relation between validation reward and expressiveness in a more granular form. The figure contains a curve for each pair (n_r, n_t) considered (in each curve, the number of symbols $|W|$ varies so as to span the interval $[0, 4]$ for the expressiveness). The validation reward for $\pi^{\text{comm}}_{\text{Opt}}$ is represented by a dot lying at $e = 1$.

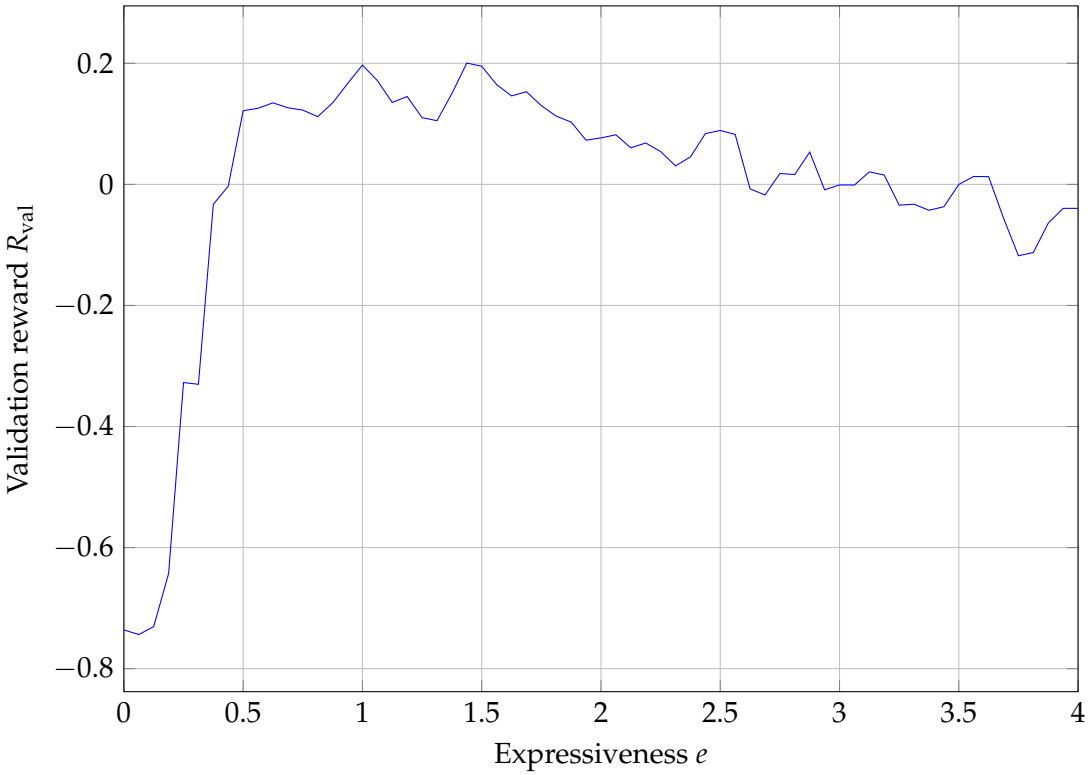


FIGURE 2.1: Normalized validation reward R_{val} vs. the expressiveness e, average across all the experiments.

The impact of expressiveness on validation reward is more evident for larger values of n_r and n_t: it can be seen that the upper lines (corresponding to lower values of n_r and n_t) tend to exhibit a constant validation reward, except for very low values of expressiveness, while the bottom lines show a degrading behavior when expressiveness grows. It can also be seen that the gap between $\pi^{\text{comm}}_{\text{Opt}}$ and the learned policies is higher for larger values of n_r and n_t.

Further insights on the relation between validation reward and expressiveness can be obtained from Table 2.2 and Figure 2.3. In particular, Table 2.2 shows that when e is sufficiently large (i.e., when $|W|$ is at least equal to $n_r n_t$) R_{val} is higher if n_t is low and viceversa. This result confirms the intuition that cooperating toward reaching less targets is easier than toward reaching more targets. On the other hand, when the language is not expressive enough (i.e., when e is too small), the resulting behavior is unpredictable in the sense that there is no intuitive relationship between $n_r n_t$ and R_{val}. Interestingly, though, Figure 2.3 shows that there is significant variability in the outcome of policy learning, more so when expressiveness is large: the first quartile of the distribution for $e = 3.0$ spans nearly the same range of values as the whole distribution.

2.5.3 Results and discussion: efficiency

In this section we assess the efficiency of the policy learning. To this end, we computed the average episode reward (the *learning reward* R_{learn}) every 512 episodes of the learning process.

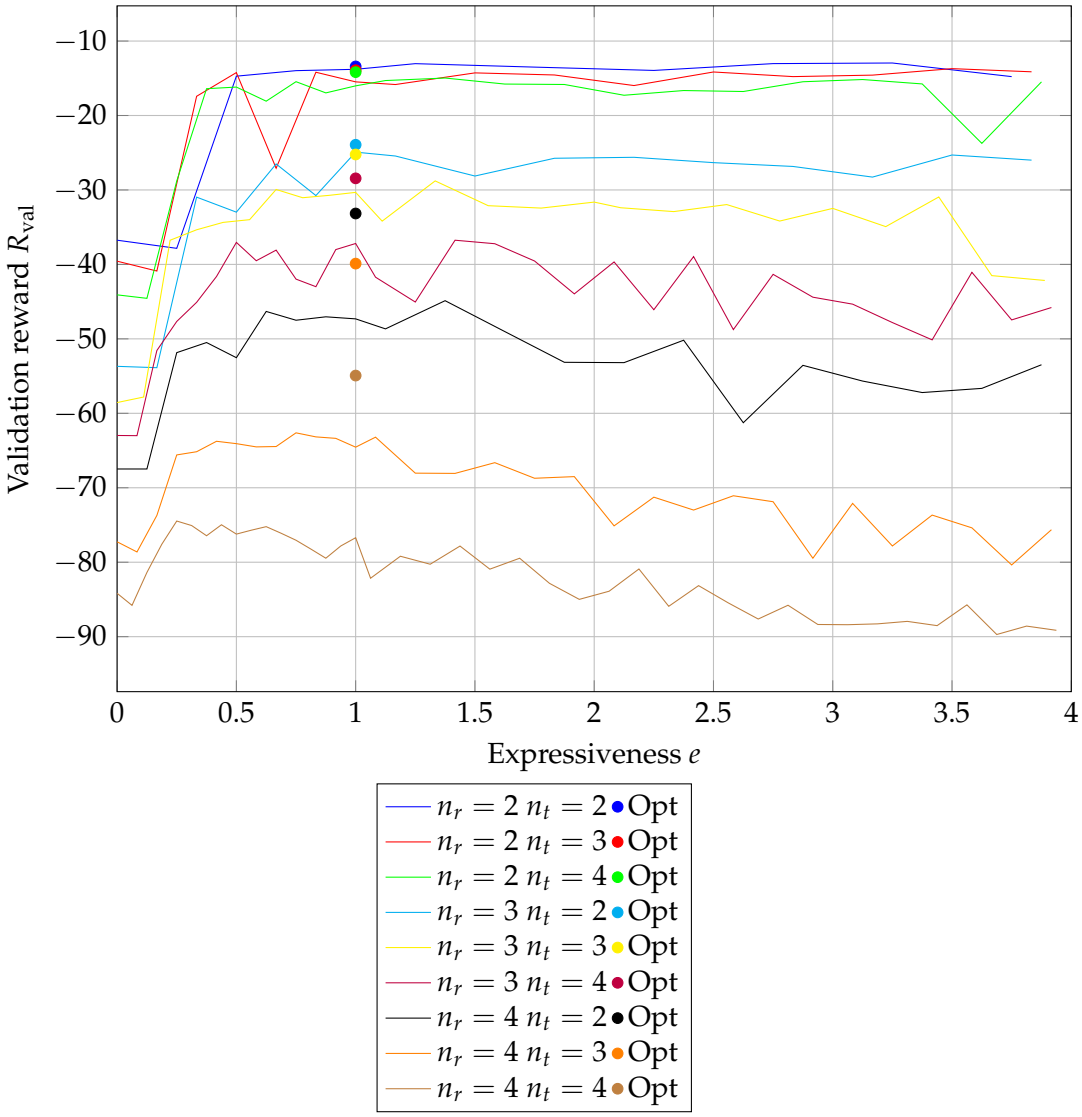


FIGURE 2.2: Validation reward R_{val} of the learned policies and of the Opt baseline vs. the expressiveness e.

TABLE 2.2: Validation reward for selected values of expressiveness.

		$e = 0.25$		$e = 1.0$		$e = 3.0$	
n_r	n_t	μ	σ	μ	σ	μ	σ
2	2	−37.84	0.10	−13.80	0.02	−13.03	0.02
	3	−40.89	0.12	−15.48	0.03	−14.78	0.03
	4	−28.80	0.03	−16.02	0.02	−15.47	0.02
3	2	−53.87	0.17	−24.92	0.08	−26.84	0.04
	3	−36.75	0.12	−30.32	0.05	−32.48	0.23
	4	−47.66	0.10	−41.72	0.10	−45.34	0.15
4	2	−51.85	0.08	−47.31	0.12	−53.56	0.12
	3	−65.58	0.08	−63.20	0.08	−72.10	0.20
	4	−74.47	0.39	−82.20	0.20	−88.40	0.09

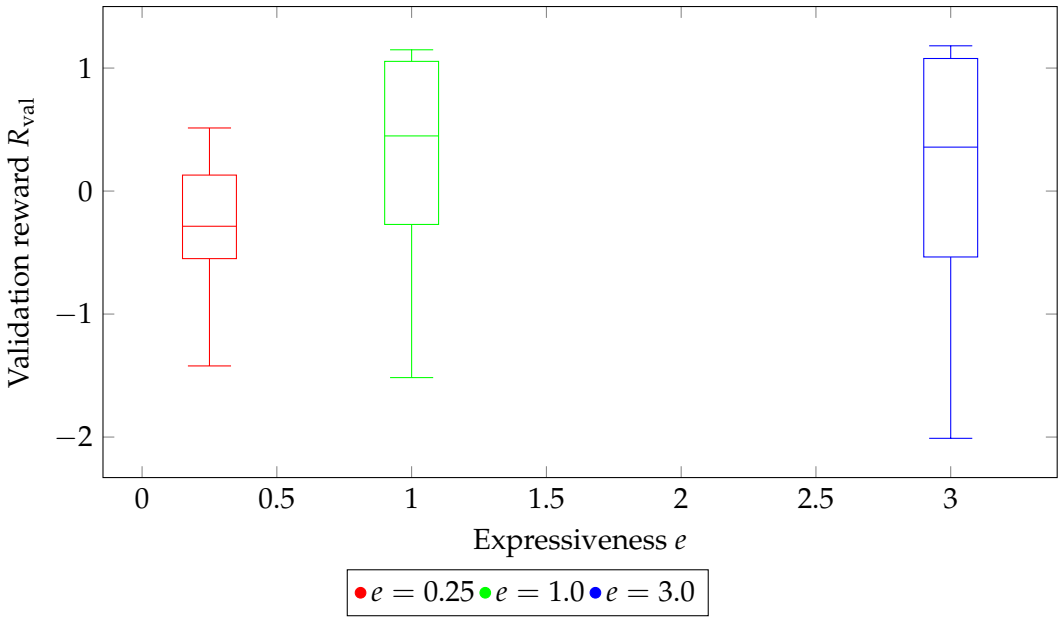


FIGURE 2.3: Boxplot of the normalized validation reward R_{val} for selected values of expressiveness e.

Figure 2.4 shows how R_{learn} changes during the learning for 3 combinations of n_r, n_t (one for each plot). For each combination we considered 3 values for expressiveness and the baseline policies.

With $n_r = n_t = 2$, all policies reach the respective maximal learning reward relatively quickly, thus very efficiently. It can also be observed a sharp separation between two groups of policies, one including $\pi^{\text{comm}}_{\text{NoComm}}$ and the learned policy with $e = 0.25$ and another including all the other policies. With $n_r = n_t = 3$, the baseline policies exhibit the same efficiency as in the previous scenario, while the learned policies require more episodes to converge to their maximal learning reward. The fact that the baseline policies converge more quickly may be explained with the fact that these policies have to learn only a movement policy. Interestingly, in this case the learned policies always exhibit a learning reward in between the two baselines. With $n_r = n_t = 4$, on the other hand, $\pi^{\text{comm}}_{\text{Opt}}$ is much slower for reaching its maximal learning reward, while all the other policies converge more quickly. We interpret this

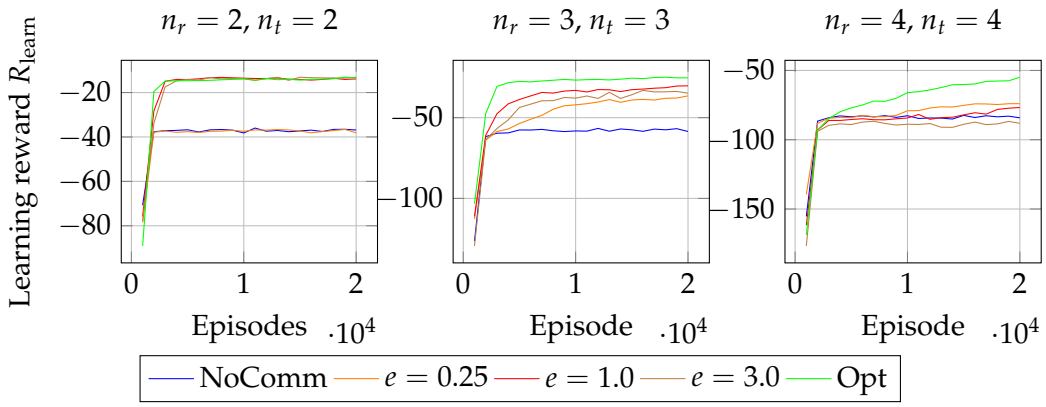


FIGURE 2.4: Learning reward R_{learn} during the learning, one curve for each of three selected expressiveness values e and each baseline, one plot for each combination of n_r, n_t.

result as a combination of the larger size of the search space coupled with the ability of $\pi^{\text{comm}}_{\text{Opt}}$ to indeed cope with such a larger space better than the other policies, i.e., still being able to learn.

Table 2.3 shows similar information of Figure 2.3 for all the experiments. For each combination of n_r, n_t and each value of e in a set of three selected values (and each of the two baselines), the table shows the number of episodes (in thousands) the learning took to reach 95% of the final value of the learning reward R_{learn}.

TABLE 2.3: Number of episodes (10^3) required for reaching 95% of the final learning reward.

		Baseline		e		
n_r	n_t	NoComm	Opt	0.25	1	3
2	2	2	3	2	3	4
	3	3	3	16	6	8
	4	3	5	13	6	8
3	2	2	3	6	6	6
	3	3	4	12	9	9
	4	3	8	11	13	15
4	2	3	3	5	10	9
	3	3	10	10	9	13
	4	3	14	12	16	3

2.6 Concluding remarks

We considered a cooperative multi-agent system in which communication between agents is required for accomplishing the task but in which agents must learn to communicate. Specifically, each agent must learn its target from what it hears, learn to move toward that target, and learn to broadcast information useful to other agents. We have considered a symbol-based language and investigated the impact of the expressiveness of the language, which is a function of vocabulary size, number of agents, and number of targets, on both effectiveness and efficiency of the learning

process. We have shown that agents including two separate neural networks, one for encoding a movement policy and another for encoding a communication policy, may indeed learn policies for moving and communicating by means of reinforcement learning. We have also shown that the best effectiveness is obtained when the vocabulary size is close to the product between number of agents and number of targets: a smaller vocabulary is not expressive enough while a larger vocabulary makes it more difficult to explore the resulting larger search space.

Chapter 3

Communication in Decision Making: Competition favors Inequality

We consider a multi-agent system in which the individual goal is to collect resources, but where the amount of collected resources depends also on others decision. Agents can communicate and can take advantage of being communicated other agents plan: therefore they may develop more profitable strategies. We wonder if some kind of collective behaviour, with respect to communication, emerges in this system without being explicitly promoted. To investigate this aspect, we design 3 different scenarios, respectively a cooperative, a competitive, and a mixed one, in which agents behaviors are individually learned by means of reinforcement learning. We consider different strategies concerning communication and learning, including no-communication, always-communication, and optional-communication. Experimental results show that always-communication leads to a collective behaviour with the best results in terms of both overall earned resources and equality between agents. On the other hand optional-communication strategy leads to similar collective strategies in some of these scenarios, but in other scenarios some agents develop individual behaviours that oppose to the collective welfare and thus result in high inequality.

3.1 Introduction

The role of autonomous machines in our society is becoming more and more important. Robotic and software agents will be performing tasks of increasing complexity with a concrete impact on our life as, e.g., autonomously delivering goods Arbanas et al., 2016 or providing feedback to learners Johnson, Gratch, and DeVault, 2017.

In complex scenarios, agents interact among themselves, constituting a multi-agent system Schatten, Ševa, and Tomičić, 2016; Calvaresi et al., 2016, and it is often the case that they may communicate to each other to better perform their task Cao et al., 2012. When agents learn, instead of being statically endowed with, their behavior, they also have to learn communication skills, resulting in the emergence of communication in the system Mordatch and Abbeel, 2018. On the other hand, single agents do not always pursue a common goal. In facts multi-agent systems may be roughly classified as cooperative, when the common goal is also the goal of single agents, competitive, when goals cannot be achieved by all agents together, along with intermediate blends. An interesting question is hence whether and how the nature of the system in terms of existence of a common goal affects the emergence of communication: do agents learn to communicate when it is useful for all of them? what if they are not directly rewarded for communicating?

In order to investigate this matter, in this paper we propose and experiment with a multi-agent system that is simple enough to allow for a detailed analysis, but tunable in terms of competition/cooperation trade-off, profitability of communication, and learnability of communication. Our system models a scenario where an agent is rewarded for accessing a resource, but the reward depends also on whether other agents are accessing the same resource, i.e., on resource occupancy. Agents may communicate their willingness to access a resource and tunable partial observability of resource occupancy makes this communication more or less important to others.

We perform several experiments on the many variants of the proposed multi-agent and analyze the outcome in terms of overall reward of the agents and inequality among agent rewards. Experimental results show that agents forced to communicate obtain the best results in terms of both overall reward and inequality in all the scenarios. When they cannot communicate, agents individually perform almost in the same way, but, in most cases, they exhibit higher inequality. Surprisingly, agents that may or may not communicate perform like those forced to communicate only in the cooperative scenario; in competitive scenarios, the overall reward of these agents is lower and the inequality is higher, despite the fact that, sometimes, some of them individually outperform the ones employing others strategies in terms of reward. These results show that collective behaviour is influenced by the scenario and the communication strategy. From our experiments we find that collective behaviour emerges in two cases: (i) in a cooperative scenario, even in presence of optional-communication strategy, and (ii) in a competitive scenario with the always-communication strategy. On the other hand, in a competitive scenario in which agents can decide whether to communicate or not, selfish behaviours appear to be more convenient, despite introducing an higher inequality.

3.2 Related works

Collective behaviour has been studied for a long time and from many points of view. However, we are not aware of any study concerned specifically on how emergence

of communication is affected by the type of collective framework (cooperative vs. competitive), which is the research question we attempt to address in this paper.

In the following sections, we briefly survey relevant works concerning scenarios and methods similar to the ones considered and used here.

3.2.1 Collective behavior

The authors of one of the seminal works Turner, Killian, et al., 1957 about collective behavior address the emergence of behaviour, considering both the environmental and the social factors, and the role of communication in it. Plenty of collective behaviour algorithms have been proposed in the literature Rossi et al., 2018, and have been extensively used in applications that require coordination algorithms. Many recent works have considered computational models for studying the emergence of collective behaviour, since they can give solutions to current real world complex problems Zhang et al., 2019, but can also be used to study future scenarios involving by intelligent machines Rahwan et al., 2019. As in Seredyński and Gąsior, 2019, in our work the agents are individually rewarded, but we investigate the collective behaviour and the overall rewards.

3.2.2 Communication

In this work, the way we define communication is inspired by consensus algorithms Ren, Beard, and Atkins, 2007: each agent shares its state with all the others and the next action is influenced by all the previous states. One relevant aspect in multi-agent systems dealing with communication is the learning of a communication protocol. Some works have treated this aspect in details Foerster et al., 2016; Mordatch and Abbeel, 2018; Talamini, Medvet, and Bartoli, 2019, and provided solutions based on differentiation of the communication error, or based on centralized learning and decentralized execution. Aware of the challenges, we do not focus on learning a way to communicate, but we investigate the impact of communication on a system-wise level.

3.2.3 Cooperative systems

Multi agent systems can be broadly divided into two groups: cooperative and competitive ones. An extensive study about collaboration in multi agent system has been presented in the cooperative framework described in Elkind, Chalkiadakis, and Wooldridge, 2012. More in detail, how agents group together in order to to improve their performance and creating coalition structure as been largely described in Rahwan et al., 2015. Differently from the works cited in Rahwan et al., 2015, in this paper we focus on the analysis of behaviours, instead of on the algorithms for efficiently creating collective structure. On the impact of communication in coordinating agents, Jaques et al., 2019 simulates alternate actions a single agent could have taken, and compute their effect on the behaviour of others; the more an action influences others in terms of changes in their behaviour, the more that action is rewarded. More on the importance of communication, Naghizadeh et al., 2019 focuses on the benefits of sharing information when agents have to coordinate, and their observations are heterogeneous. Concerning the problem of minimizing the amount of communication required for coordinating agents, in Zhang and Lesser, 2013 for instance, agents are allowed to learn to dynamically identify whom to coordinate with. This constraint is

not explicitly presented to the agents in our work, but the amount of communication is directly influenced by the individual objective.

3.2.4 Competitive systems

A different type of multi-agent systems are those defined as competitive, in which agents rival against each other. Authors of a recent study Singh, Jain, and Sukhbaatar, 2018 claim that multi-agent systems other than cooperative ones, namely competitive or mixed, have not been extensively studied. As for the cooperative ones, a key aspect in competitive system is the communication between agents, in particular the trust model, which define how and when to trust the information obtained from another agent Yu et al., 2013. In McPartland, Nolfi, and Abbass, 2005 agents are divided in teams where teammates are allowed to communicate and teams are given a competitive task. Results show that communication improves team overall strategy. Again on emergence of communication in competitive scenarios, Lehman et al., 2018 robots were evolved to find food sources while avoiding poison. In some cases, when robots adapted to understand blue as a signal of food, competing robots evolved to signal blue at poison instead. In other experiments robots literally hide the information from others. Overall, a simple on-off switch for communication revealed a surprisingly rich evolutionary potential. In Bansal et al., 2017 the authors prove that emergence of complex behaviour does not require a complex environment, but can be induced by having learning agents competing in the same scenario.

3.2.5 Reinforcement learning

An extensive study on the different machine learning techniques available for optimization in multi-agent systems, including both single learners and multiple simultaneous learners, has been reported in Panait and Luke, 2005. Among these techniques, Reinforcement Learning (RL) has recently attracted a lot of interest, due to the outstanding results of Mnih et al., 2015; Silver et al., 2016 and the recent advances of Deep Learning. Mostly for this reason, we chose to use RL as the algorithm under the hood of the agent learning. RL has been extensively applied for finding optimal policies in multi-agent systems, from Littman, 1994; Tan, 1993 to more recently Leibo et al., 2017; Shoham, Powers, and Grenager, 2007, and also when communication is involved Talamini, Medvet, and Bartoli, 2019. In this context, the same policy may be employed for all the agents, if diversity of behaviour and social aspects are not relevant aspects. On the contrary, when considering independent policies for all the agents, different role may be assigned to agents, like in Ahilan and Dayan, 2019, where the authors consider a hierarchical society with a manager and subordinates, or like in Zhang, Lesser, and Abdallah, 2010, where to improve the learning speed, multi-agent RL is managed by a hierarchical organization, where groups of interacting agents learning together are coordinated by supervisors. Differently from Ahilan and Dayan, 2019; Zhang, Lesser, and Abdallah, 2010, in this work we treat agents with the same role and as independent learners, meaning that every agent has its own policy Tan, 1993. When employing independent learners, due to the continually changing policies of agents during training Papoudakis et al., 2019, most of the RL algorithms incur into non-stationarity issues, that make the training more difficult. This problems have been tackled in Lowe et al., 2017 by introducing a centralized entity, that helps stabilizing the training.

3.2.6 Inequality

For what concerns inequality, many works have been proposed to study how to mitigate this phenomenon, and to promote altruistic behaviour. In Hughes et al., 2018; Mazzolini and Celani, 2020; Wang et al., 2019 the authors consider multi-agent systems, in which agents learn optimal behaviour, subjected to a trade-off between the short-term individual reward and long-term collective interest. The emergence of inequality-averse behaviour depends on the environment-related aspects, like the abundance of resources, as highlighted in Mazzolini and Celani, 2020. For contrasting inequality, Hughes et al., 2018 introduces the possibility for an agent to punish another one. Results show that most of the agents end up developing inequality-aversion behaviours, and the pro-social agents punish the defectors. Finally, in Wang et al., 2019 the cooperation is promoted by sharing among the agents the same reward function, that is optimized to maximize the collective return of the agents. With respect to previous articles, in this work we identify mandatory communication for all the agents as a valuable countermeasure for contrasting inequality.

3.3 Model

3.3.1 System state

We consider a discrete time dynamic multi-agent system with n_a agents and n_r resources.

We denote by $\boldsymbol{R}^{(t)} = \left(R_1^{(t)}, \dots, R_{n_r}^{(t)}\right)$ the distribution of agents on resources at time t, where $R_i^{(t)} \subseteq \{1, \dots, n_a\}$ is the set of agents accessing i-th resource. It holds that $\forall i, j, t : R_i^{(t)} \cap R_j^{(t)} = \emptyset$, i.e., each agent can access at most one resource at the same time. We say that an agent is *inactive* when it is not accessing any resource: i.e., if $\forall i : R_i^{(t)} \not\ni j$, then the j-th agent is inactive at time t. Consequently, we say that the j-th agent is *active* if $\exists i : R_i^{(t)} \ni j$. We call the *filling* of the i-th resource the number $\rho_i^{(t)} = \left|R_i^{(t)}\right|$ of agents accessing that resource at a given time.

We denote by $\boldsymbol{U}^{(t)} = \left(U_1^{(t)}, \dots, U_{n_a}^{(t)}\right)$ the agent states, where $U_i^{(t)} = \left(u_i'^{(t)}, u_i''^{(t)}\right)$ is the i-th agent state at time t. The i-th agent state is a pair composed by $u_i'^{(t)} \in \{1, \dots, n_r\} \cup \{\perp\}$, that defines the resource the i-th agent is active at time t, and $u_i''^{(t)} \in \{1, \dots, n_r\} \cup \{\perp\}$, that defines the i-th agent *preference* between resources at time t. Intuitively, $u_i'^{(t)}$ is where the agent is and $u_i''^{(t)}$ is where the agent wants to go.

Communication consists in a tuple $W_i^{(t)} = (w_i'^{(t)}, w_i''^{(t)})$, which we call *word*, emitted by each agent at each time step and heard by all the other agents. We denote by $\boldsymbol{W}^{(t)} = \left(W_1^{(t)}, \dots, W_{n_a}^{(t)}\right)$ the words emitted at time t. The communication $W_i^{(t)}$ emitted by i-th agent at time t is composed by $w_i'^{(t)} \in \{1, \dots, n_r\} \cup \{\perp\}$ and by $w_i''^{(t)}) \in \{1, \dots, n_r\} \cup \{\perp\}$: the semantics of $\boldsymbol{W}^{(t)}$ is the same one of $\boldsymbol{U}^{(t)}$.

Given these definitions, the system state at time t is described by:

$$s^{(t)} = \left(\boldsymbol{R}^{(t)}, \boldsymbol{U}^{(t)}, \boldsymbol{W}^{(t)}\right) \tag{3.1}$$

At the initial time $t = 0$, the system state is $s^{(0)} = \left(\boldsymbol{R}^{(0)}, \boldsymbol{U}^{(0)}, \boldsymbol{W}^{(0)}\right)$, with $\boldsymbol{R}^{(0)} = \{\emptyset, \dots, \emptyset\}$, $\boldsymbol{U}^{(0)} = \{\{\perp, \perp\}, \dots, \{\perp, \perp\}\}$ and $\boldsymbol{W}^{(0)} = \{\{\perp, \perp\}, \dots, \{\perp, \perp\}\}$. That

is, all the agents are inactive, they have all default initial state, and no words have been spoken so far.

3.3.2 System dynamics

Observation

The agents do not have full knowledge of the system state. Instead, i-th agent observes—i.e., knows—the filling of the resources $\boldsymbol{\rho}^{(t)}$, its own state $U_i^{(t)}$, and an aggregate $\boldsymbol{V}^{(t)}$ of the words $\boldsymbol{W}^{(t)}$ spoken at previous time step. In particular, the i-th agent at time t observes $\boldsymbol{V}^{(t)} = \left(v_1^{(t)}, \dots, v_{n_r}^{(t)}\right)$, where:

$$\begin{aligned} v_j^{(t)} = \sum_{i=1}^{n_a} (n_a + 1) \mathbb{I}\left(w_i'^{(t)} = w_i''^{(t)} = j\right) \\ + \mathbb{I}\left(w_i'^{(t)} \neq w_i''^{(t)} \wedge w_i''^{(t)} = j\right) \end{aligned} \tag{3.2}$$

where $\mathbb{I} : \{\text{true}, \text{false}\} \to \{0, 1\}$ is an indicator function. Intuitively $v_j^{(t)}$ is a weighted sum of the number of agents accessing, and willing to access, the j-th resource and the number of agents not currently accessing but willing to access the same j-th resource. In other words, $v_j^{(t)}$ acts as a predictor for $\rho_i^{(t+1)}$.

Formally, the information available to the i-th agent at time t is a triplet $o_i^{(t)} = \left(\boldsymbol{\rho}^{(t)}, U_i^{(t)}, \boldsymbol{V}^{(t)}\right)$, therefore $o_i \in O = \{1, \dots, n_a\}^{n_r} \times (\{1, \dots, n_r\} \cup \{\perp\})^2 \times \{1, \dots, n_a(n_a + 1)\}^{n_r}$.

Action

Every i-th agent at time t, can take an action $a_i^{(t)} = \left(a_i'^{(t)}, a_i''^{(t)}\right)$, where $a_i'^{(t)} \in \{0, \dots, n_r\}$, controls which resource to set the preference to, and $a_i''^{(t)} \in \{0, 1\}$ controls whether to communicate or not, therefore $a_i^{(t)} \in A = \{1, \dots, n_r\} \times \{0, 1\}$.

Actions change the state of the system as follows. The i-th agent state $U_i^{(t+1)}$ at time $t + 1$, is updated as:

$$u_i'^{(t+1)} = \begin{cases} a_i'^{(t)} & \text{if } a_i'^{(t)} = u_i''^{(t)} \\ u'^{(t)} & \text{otherwise} \end{cases} \tag{3.3}$$

$$u_i''^{(t+1)} = a_i'^{(t)} \tag{3.4}$$

That is, the agent changes the resource it is accessing only if it confirms its previous preference; the preference itself is always updated. The i-th agent word $W_i^{(t+1)}$ emitted at time t, is updated as:

$$W_i^{(t+1)} = \begin{cases} U_i^{(t+1)} & \text{if } a_i''^{(t)} = 1 \\ \perp & \text{otherwise} \end{cases} \tag{3.5}$$

That is, the agent emits a word only if the second element of the action pair is 1.

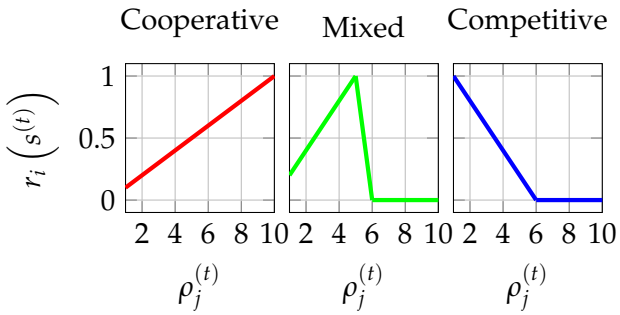


FIGURE 3.1: Reward functions in 3 scenarios.

Policy

Agents take actions according to their policy function. The i-th agent policy at time t can be any function that outputs an action $a_i^{(t)} \in A$, given the current agent observation $o_i^{(t)} \in O$.

Reward

We define a reward function for the i-th agent, and the system in state $s^{(t)}$ at time t, as $r_i(s^{(t)}) : [0, n_a] \to [0, 1]$. We differentiate the actual form of $r_i(s^{(t)})$ depending on the kind of scenario, i.e., cooperative, competitive, or mixed, as follows:

$$r_{i,\text{coop}}\left(s^{(t)}\right) = \frac{\rho_j^{(t)}}{n_a n_r} \tag{3.6}$$

$$r_{i,\text{comp}}\left(s^{(t)}\right) = \max\left(0, \frac{\frac{n_a}{n_r} + 1 - \rho_j^{(t)}}{\frac{n_a}{n_r}}\right) \tag{3.7}$$

$$r_{i,\text{mixed}}\left(s^{(t)}\right) \begin{cases} \frac{\rho_j^{(t)}}{n_a n_r} & \text{if } \rho_j^{(t)} \leq \frac{n_a}{n_r} \\ \max\left(0, \frac{n_a}{n_r} - \rho_j^{(t)} + 1\right) & \text{otherwise} \end{cases} \tag{3.8}$$

where $j = u_i'^{(t)}$ is the index of the resource being accessed by the i-th agent. Figure 3.1 shows the plots for the three considered reward functions. The semantics is clearly visible in the shape of each plot. The reward based on the resource filling is: the most crowded, the better, for the cooperative scenario; the least crowded, the better, for the competitive scenario; and the closer to the optimal capacity, the better, for the mixed scenario.

The goal of the game is to find for every i-th agent, the policies that maximize its individual reward starting from time t_0, for a number T_{episode} of time steps, defined as:

$$J_i^{(t_0)} = \sum_{t=t_0}^{t=t_0+T_{\text{episode}}} r_i(s^{(t)}) \tag{3.9}$$

The overall reward J for a group of n_a agents with individual rewards $J_i, \ldots, J_{n_a}$ is therefore:

$$J^{(t_0)} = \frac{1}{n_a} \sum_{i \in \{1,\ldots,n_a\}} J_i^{(t_0)} \tag{3.10}$$

The inequality I for a group of n_a agents with individual rewards $J_1, \ldots, J_{n_a}$ is the standard deviation of individual rewards:

$$I^{(t_0)} = \sqrt{\frac{1}{n_a - 1} \sum_{i=1}^{n_a} \left(J_i^{(t_0)} - J^{(t_0)} \right)^2} \tag{3.11}$$

3.3.3 Policy learning

We consider agents as independent learners and, since both the observation space O and the action space A are discrete, we do not use function approximates. Each agent is given a sparse tabular policy characterized by *state-action value function* $Q_i : O \times A \mapsto \mathbb{R}$.

At time t the i-th agent picks action $a_i^{(t)}$ using an ϵ-greedy policy, given $p \sim U([0,1])$, and exploration probability ϵ^k after k training iterations, defined as:

$$a_i^{(t)} = \begin{cases} \arg\max_{a \in A} Q_i^k \left(o_i^{(t)}, a \right) & \text{if } p < \epsilon^k \\ a \sim U(A) & \text{otherwise} \end{cases} \tag{3.12}$$

where $U(A)$ is the uniform distribution over A. At the initial training iteration k_0, $\forall i \in \{1, \ldots, n_a\}, \forall t \in t_0, \ldots, T_{\text{episode}}, \forall o_i^{(t)} \in O, \forall a \in A, Q_i^{k_0} \left(o_i^{(t)}, a \right) = 0$.

We perform policy learning of the values stored in $Q_1, \ldots, Q_{n_a}$ by means of Q-learning Watkins and Dayan, 1992. For every k-th training iteration, the i-th agent state-action value function Q_i is updated with learning rate α and discount factor $\gamma \in [0,1]$, as:

$$\begin{aligned} Q_i^k \left(o_i^{(t)}, a_i^{(t)} \right) = & Q_i^{k-1} \left(o_i^{(t)}, a_i^{(t)} \right) + \alpha \left(r_i \left(s^{(t)} \right) \right. \\ & \left. + \gamma \max_{a \in A} Q_i^{k-1} \left(o_i^{(t+1)}, a \right) - Q_i^{k-1} \left(o_i^{(t)}, a_i^{(t)} \right) \right) \end{aligned} \tag{3.13}$$

For every k-th training iteration, exploration rate for every agent is exponentially decreased from ϵ_i to ϵ_f with decay δ_ϵ.

3.4 Experiments

We wanted to investigate if and how the emergence of communication is impacted by the type of collective framework. To this end, we explored 3 different strategies concerning the communication part of the action. Two of them do not allow the agent to choose: *no-comm*unication, i.e., $a_i''^{(t)} = 0$, and *always*-communication, i.e., $a_i''^{(t)} = 1$. One, that we call the *optional*-communication strategy, allows to choose and the actual way of choosing is learned. Finally, as a baseline we considered a fourth case in which the entire policy of the agent is *random*, instead of being learned, i.e., $a_i^{(t)} \sim U(A)$.

We consider also the non-observable filling variation for each scenario, that is, a scenario in which the filling is not available to the agents, i.e., $o_i^{(t)} = \left(U_i^{(t)}, V^{(t)} \right)$. Intuitively, this variant is important because communicating is the only way, for the agents, to know where it is convenient to go.

For each combination of collective framework (cooperative, competitive, mixed), each strategy (always, optional, no-comm, random), and each variant (with filling in o, without), we performed n_{trials} training lasting n_{train} episodes. Table 6.1 shows the training parameters used in the experimental campaign.

TABLE 3.1: Experiments parameters.

	Parameter	Value
Sim.	Trials n_{trials}	20
	Training episodes n_{train}	20 000
	Validation episodes n_{val}	100
	Episode time steps T_{episode}	100
	Number of agents n_a	10
	Number of resources n_r	2
Agent	Initial exploration rate ϵ_i	1.0
	Final exploration rate ϵ_f	0.01
	Exploration decay δ_ϵ	0.9995
	Learning rate α	0.1
	Discount factor γ	0.9

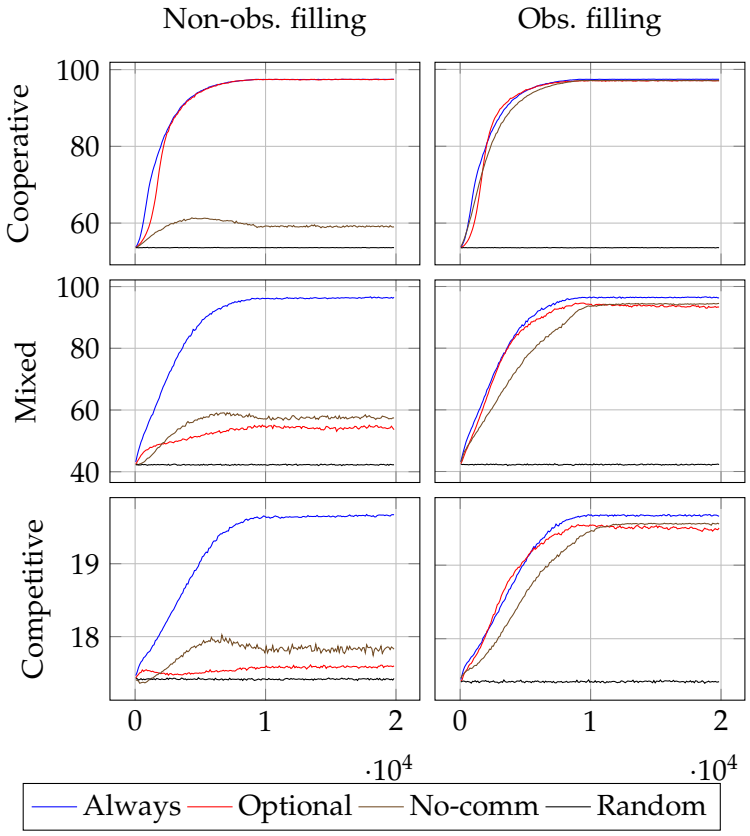


FIGURE 3.2: Overall reward J over n_{train} training episodes.

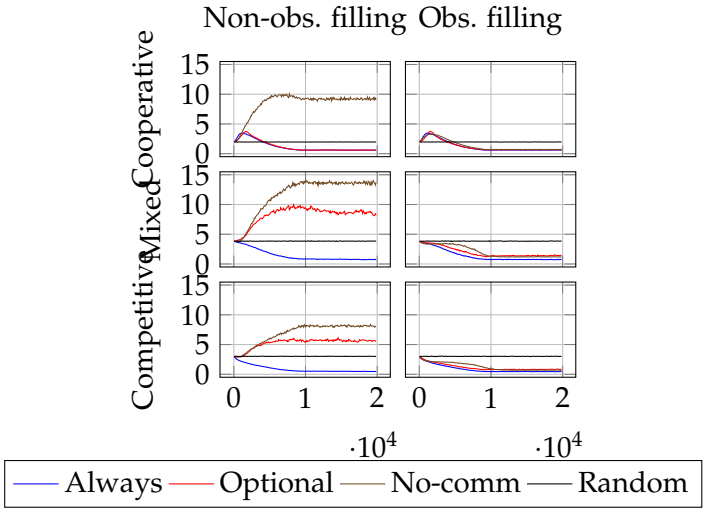


FIGURE 3.3: Inequality I over n_{train} training episodes.

3.4.1 Strategies effectiveness

Figures 3.2 and 3.3 show the training results. From these figures it can be seen that agents employing always-communication strategy achieve the best overall reward J, and the lowest inequality I among agents. This result confirms that indeed communication is needed for reaching the best overall results.

The importance of communication is more evident in scenarios with non-observable filling, where agents can only rely on what they listen, in order to gain information on the system state. In these scenarios, the gap between always-communication and no-communication strategy in terms of overall reward and inequality is more noticeable. Considering observable filling cases, there is still an advantage of always-communication strategy in terms of overall reward J and inequality I with respect to the no-communication strategy.

Optional-communication strategy results lay in between the always-communication and no-communication ones, depending on the scenario considered. This strategy achieves best overall reward in the cooperative scenarios, even with non-observable filling, where the emergence of communication is needed. In this scenario optional-communication are equally good as always-communication in terms of both overall reward and inequality. On the other hand this strategy performs poorly in the competitive and mixed scenarios with non-observable filling, in terms of both overall reward and inequality.

Finally, it is important to note that the random strategy is always worst than all the other ones, both in terms of overall reward J and inequality I: the learning helps agents to make better decision than choosing at random, even if in presence of a no-communication strategy.

3.4.2 Strategies efficiency

In order to measure how far is the current resources filling from the uniform distribution of agents, we introduce a distance we call *displacement*. For n_a agents and n_r resources, we define the *displacement* $d^{(t)}$ at time t, averaged on n_{val} validation episodes, as:

$$d^{(t)} = \frac{1}{n_{\text{val}}} \sum_{e=1}^{n_{\text{val}}} \sum_{j=1}^{n_r} \left| \rho_j^{(t)} - \frac{n_a}{n_r} \right| \tag{3.14}$$

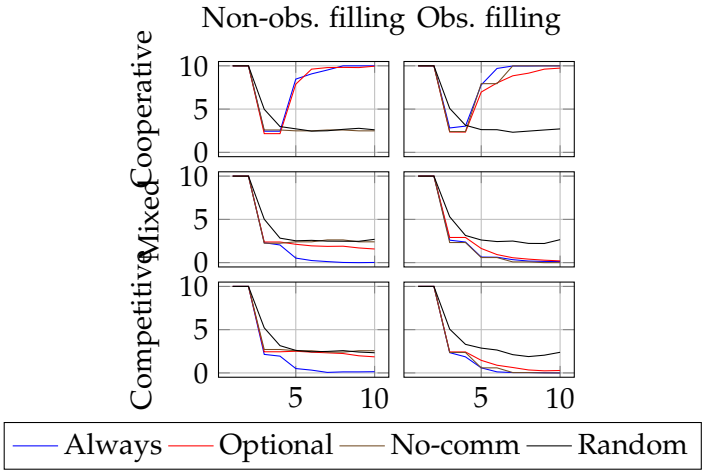


FIGURE 3.4: Resources displacement d during the first 10 steps of validation.

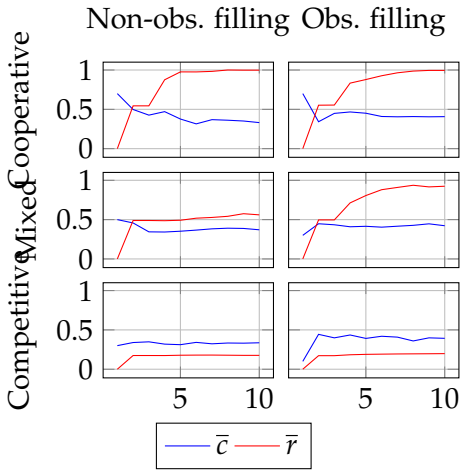


FIGURE 3.5: Average communication $\bar{c}$ and average reward $\bar{r}$ for optional-communication strategy in n_{val} validation episodes.

In Figure 3.4 we show the displacement $d^{(t)}$, with $t = 1, \ldots, 10$, and $n_{\text{val}} = 100$ episodes, where each line represents a different strategy employed; we compute the displacement for all the scenarios.

In competitive and mixed scenarios, with n_a agents and n_r resources, the overall optimal displacement at time t is $d^{(t)} = 0$, that is the agents are uniformly active on the resources. Differently in the cooperative scenario, with n_a agents and n_r resources, the overall optimal displacement at time t is $d^{(t)} = n_a$, that is the agents are all active on 1 resource.

From Figure 3.4 we can see that also in validation episodes the always-communication strategy is not only the most effective, but also the most efficient collective strategy for reaching the overall optimal displacement value in all the scenarios. Also from this figure we can see that agents displacement converges to the final value within the first 10 steps of an episode, regardless of the strategy considered. Motivated by this finding, in Figure 3.5 and tables 3.3 and 3.4 we consider only the first 10 steps of validation episodes.

3.4.3 Optional-communication results

Collective considerations

Given n_a agents, n_{val} validation episodes, we denote the average communication $\overline{c}^{(t)}$ at time t as:

$$\overline{c}^{(t)} = \frac{1}{n_a n_{\text{val}}} \sum_{e=1}^{n_{\text{val}}} \sum_{i=1}^{n_a} a_i''^{(t)} \tag{3.15}$$

Given n_a agents, n_{val} validation episodes, we denote the average reward $\overline{r}^{(t)}$ at time t as:

$$\overline{r}^{(t)} = \frac{1}{n_a n_{\text{val}}} \sum_{e=1}^{n_{\text{val}}} \sum_{i=1}^{n_a} r_i(s^{(t)}) \tag{3.16}$$

In Figure 3.5 we show the average communication $\overline{c}^{(t)}$ and the average reward $\overline{r}^{(t)}$, for $t = 1, \ldots, 10$ steps of n_{val} validation episodes. From the same figure, it can be seen that in the cooperative scenarios, the average communication $\overline{c}$ has is higher in the first couple of steps, and the amount of communication provided in these steps is sufficient to achieve nearly maximum average reward $\overline{r}$ in few steps. This means that optional-communication agents can achieve always-communication level results in cooperative scenarios, in particular with non-observable filling, in terms of both overall reward (Figure 3.2) and inequality (Figure 3.3) during the training phase, and from the validation (Figure 3.5) we can see that a smaller amount of communication is needed to perform like always-communication agents.

Individual considerations

In Figure 3.6 we show the distribution of individual validation reward for respectively the always-communication, optional-communication, and no-communication strategy. Table 3.2 reports the maximum reward value reached for each scenario in the validation. From Figure 3.6 appears that the competitive observable filling scenario is the most interesting to us: in some validation episodes, it occurs that few agents employing optional-communication strategy achieve higher individual reward with respect to the majority of the agents. Moreover, these agents in this scenario outperform agents employing any other strategy in terms of individual reward. In other words, if considering overall reward (Figure 3.2) optional-communication performs poorly in this scenario, but from an individual point of view, the single agents achieve highly unbalanced rewards.

Tables 3.3 and 3.4 show the sequence of actions of n_a optional-communication agents during the first 10 steps of a validation episode, respectively in the cooperative non-observable filling scenario (Table 3.3), and in the competitive observable filling scenario (Table 3.4). Here we aim to capture relevant information on the system state and agents policy by introducing a simpler notation: we consider the i-th agent state $U_i^{(t)}$ at time t, we say that the i-th agent changes its preference if: $u_i'^{(t)} \neq u_i'^{(t-1)}$. In the same way we say that the i-th agent communicates at time t if: $a_i''^{(t)} = 1$. In this table, the i-th agent confirming its preference for resources at time t is indicated by the $\bigcirc$ symbol in i-th line of the table, at the t-th position of the sequence of actions, regardless of its color. The i-th agent changing its state, by setting its preference to resource 1 (or similarly over resource 2) at time t is indicated by the $\bigtriangledown$ symbol (or similarly the $\bigtriangleup$ symbol) in the i-th line of the table, at the t-th position of the sequence of actions, regardless of its color. The i-th agent is active on resource 1 (or similarly on resource 2) at time t, when on the i-th line of the table, the symbol $\bigtriangledown$

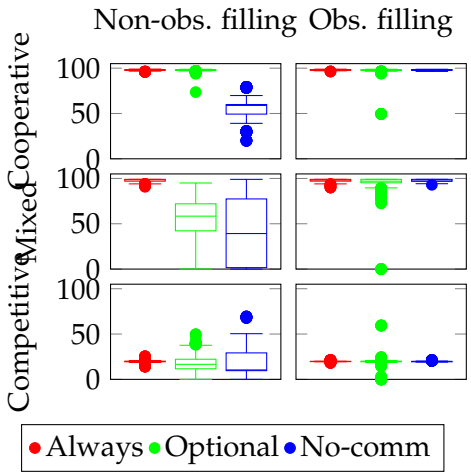


FIGURE 3.6: Individual reward distribution in validation episodes.

TABLE 3.2: Max validation return.

	Agent	Non-obs. filling	Obs. filling
Coop.	Always	99.0	99.0
	No-comm	79.3	99.0
	Optional	98.8	99.0
Mixed	Always	99.0	99.0
	No-comm	89.0	99.0
	Optional	93.6	95.0
Comp.	Always	25.4	21.6
	No-comm	69.4	21.4
	Optional	49.8	59.4

TABLE 3.3: Optional-communication policies in a cooperative non-observable filling scenario.

Agent	Actions
1	△ ○ ▽ ○ ○ ○ △ ○ ○ ▽
2	△ ○ ▽ ○ ○ △ ○ ○ ○ ○
3	▽ ○ ○ ○ ○ △ ▽ △ ○ ○
4	△ ○ ○ ○ ○ ○ ▽ ○ ○ ○
5	△ ○ ▽ ○ ○ ○ ○ △ ○ ○
6	▽ ○ ○ ○ ○ △ ▽ △ ○ ○
7	△ ○ ▽ ○ ○ ○ ○ △ ○ ○
8	▽ ○ ○ ○ ○ △ ○ ○ ○ ○
9	△ ○ ○ ○ ○ ▽ △ ▽ ○ ○
10	▽ ○ ○ ○ ○ △ ○ ○ ○ ○

○ confirm, no-comm.
○ confirm, comm.
▽ change to 1, no-comm.
▽ change to 1, comm.
△ change to 2, no-comm.
△ change to 2, comm.

TABLE 3.4: Optional-communication policies in a competitive observable filling scenario.

Agent	Actions
1	△ ○ ○ ○ ○ ○ ▽ △ ○ ○
2	▽ ○ ○ ○ ○ △ ▽ ○ △ ▽
3	▽ ○ ○ ○ ○ ○ ○ △ ○ ○
4	▽ ○ △ ▽ ○ ○ △ ▽ ○ △
5	△ ○ ▽ △ ○ ○ ○ ▽ △ ▽
6*	▽ ○ △ ▽ ○ △ ▽ ○ ○ ○
7	△ ○ ▽ △ ▽ △ ○ ▽ △ ▽
8	△ ○ ○ ▽ △ ○ ○ ○ ▽ △
9	▽ ○ ○ ○ ○ △ ▽ △ ▽ △
10	▽ ○ ○ ○ ○ △ ○ ○ ○ ▽

(or similarly △) is at the $t-1$-th position of the sequence of actions, immediately followed by the symbol ○ at the t-th position, regardless of their color. The i-th agent is communicating its state at time t, when on the i-th line of the table, the symbol at the t-th position of the sequence of actions is *green*. The i-th agent communicating while not changing preference at time t, is indicated by the ○ symbol in the i-th line of the table, at the t-th position of the sequence of actions, at time $t+1$ will be active on the same resource, regardless of its next decision at time $t+1$. Differently, the i-th agent communicating while changing preference for resource 1 (or similarly for resource 2) at time t, is indicated by the ▽ symbol (or similarly the △ symbol) in the i-th line of the table, at the t-th position of the sequence of actions, at time $t+1$ will be active on a different resource depending on its next decision at time $t+1$.

From Table 3.3 we can see that agents seem to have learned that communicating while confirming their preference, denoted by ○, is more helpful for the listeners, rather than communicating while changing their preference, denoted by △ or ▽. This finding is supported by the high number of ○ symbol, in contrast with the low number of △ and ▽.

In Table 3.4 we show actions taken by optional-communication agents in observable filling competitive scenario during the first 10 steps of a validation episode. We indicate with $*$ the agent collecting the highest individual reward in this episode. In

this case agents seem to learn that communicating while changing their preference, for instance from resource 1 to resource 2, denoted by △, gives ambiguous information for the listeners, and therefore it can be used to trick the others. On the other side, ○ is less frequently used, since it would give useful information to the competitors. Also agents communicate less frequently, and change preference more often than in the cooperative case. This finding is supported by the high number of △ and ▽ symbols.

3.5 Concluding remarks

We considered a multi-agent system in which communication among agents is required for learning the system-wise best policies. We investigated the role of communication in the emergence a of collective behaviour in such system, by designing 3 scenarios in which different strategies are employed by agents, and where agent policies are learned by means of reinforcement learning. The experimental results show that communication is, in general, a way for reducing inequality. Moreover, agents with optional communication capabilities develop a collective behaviour in cooperative scenarios, whereas in competitive scenarios they exhibit a selfish behaviour that leverages on communication to promote their individual goal and thus resulting in high inequality.

Future ideas for expanding this study, in particular in the competitive scenarios, include the possibility of enforcing some kind of regulation (e.g., as in Lombardi, Medvet, and Bartoli, 2017) for softening selfish tendency after the learning, or employing some form of reward shaping Devlin and Kudenko, 2016 (or equivalent strategies in non-RL learning of the agent policy Talamini et al., 2018) during the learning, in order to discourage selfish behaviors.

Chapter 4

On the Impact of the Rules on Autonomous Drive Learning

Autonomous vehicles raise many ethical and moral issues that are not easy to deal with and that, if not addressed correctly, might be an obstacle to the advent of such a technological revolution. These issues are critical because autonomous vehicles will interact with human road users in new ways and current traffic rules might not be suitable for the resulting environment. We consider the problem of learning optimal behavior for autonomous vehicles using Reinforcement Learning in a simple road graph environment. In particular, we investigate the impact of traffic rules on the learned behaviors and consider a scenario where drivers are punished when they are not compliant with the rules, i.e., a scenario in which violation of traffic rules cannot be fully prevented. We performed an extensive experimental campaign, in a simulated environment, in which drivers were trained with and without rules, and assessed the learned behaviors in terms of efficiency and safety. The results show that drivers trained with rules enforcement are willing to reduce their efficiency in exchange for being compliant to the rules, thus leading to higher overall safety.

4.1 Introduction

In recent years, autonomous vehicles have attracted a lot of interest from both industrial and research groups Howard and Dai, 2014; Skrickij, Sabanovic, and Zuraulis, 2020. The reasons for this growth are the technological advancement in the automotive field, the availability of faster computing units, and the increasing diffusion of the so-called Internet of Things. Autonomous vehicles collect a huge amount of data from the vehicle and from the outside environment, and are capable of processing these data in real-time to assist decision-making on the road. The amount of collected information and the need for real-time computing make the design of the driving algorithms a complex task to carry out with traditional techniques. Moreover, the sources of information may be noisy or may provide ambiguous information that could therefore negatively affect the outcome of the driving algorithm. The combination of these factors makes it very hard, if not unfeasible, to define the driver behavior by developing a set of hand-crafted rules. On the other side, the huge amount of data available can be leveraged by suitable machine learning techniques. The rise of deep learning in the last decade has proven its power in many fields, including self-driving cars development, and enabled the development of machines that take actions based on images collected by a front camera as the only source of information Bojarski et al., 2016, or even using a biological inspired event-driven camera Maqueda et al., 2018.

The use of simulations and synthetic data Sharifzadeh et al., 2016 for training have allowed to assess neural networks capabilities in many different realistic environments and different degrees of complexity. Many driving simulators have been designed, from the low-level ones that allow the drivers to control the hand brake of their car Jaritz et al., 2018, to higher-level ones, in which the drivers can control their car acceleration and lane-change Bouton et al., 2019. Some simulators model the traffic in an urban road network Wang, Liu, and Xu, 2020, some others model car's intersection access Qiao et al., 2018; Tram et al., 2018; Liebner et al., 2012; Isele et al., 2017, or roundabout insertion Capasso, Bacchiani, and Molinari, 2020.

In a near future scenario, the first autonomous vehicles on the roads will have to make decisions in a mixed traffic environment. Autonomous vehicles will have to be able to cope with radically different road agents, i.e., agents powered by machines capable of processing information way more quickly than human drivers and human drivers that could occasionally take unexpected actions. There will hardly be a single authority to control each car in a centralized fashion and thus every autonomous vehicle will have to take decisions on its own, treating all the other road agents as part of the environment. It may very well be the case that current traffic rules do not fit a scenario with self-driving cars.

In this work, we investigate to which extent the traffic rules affect the drivers optimization process. The problem of finding the optimal driving behavior subjected to some traffic rules is highly relevant because it provides a way to define allowed behaviors for autonomous drivers, possibly without the need to manually craft those behaviors. A first approach for solving this problem consists of defining hard constraints on driver behavior and replacing forbidden actions with fallback ones Shalev-Shwartz, Shammah, and Shashua, 2016. Such an approach leads to drivers which are not explicitly aware of the rules. If those hard constraints were removed, driver behavior could change in unpredictable ways. Another approach consists in punishing behaviors that are not compliant with the rules, thus discouraging drivers from taking those behaviors again. In this work, we investigate this second approach based on punishing undesired behaviors. In this scenario, drivers have to learn the optimal behavior that balances a trade-off between being compliant with the rules and driving fast while avoiding collisions. A scenario in which drivers have the chance of breaking the rules is particularly relevant because it could address the complex ethics issues regarding self-driving cars in a more flexible way (those issues are fully orthogonal to our work, however).

We perform the optimization of the self-driving controllers using Reinforcement Learning (RL), which is a powerful framework used to find the optimal policy for a given task according to a trial-and-error paradigm. In this framework, we consider the possibility of enforcing traffic rules directly into the optimization process, as part of the reward function. Experimental results show that it is therefore possible to reduce unwanted behaviors with such approach.

4.2 Related Works

The rise of Reinforcement Learning (RL) Sutton and Barto, 2018 as an optimization framework for learning artificial agents, and the outstanding results of its combination with neural networks Mnih et al., 2013, have recently reached many new grounds, becoming a promising technique for the automation of driving tasks. Deep learning advances have proved that a neural network is highly effective in automatically extracting relevant features from raw data Krizhevsky, Sutskever, and Hinton, 2012,

as well as allowing an autonomous vehicle to take decisions based on information provided by a camera Bojarski et al., 2016; Maqueda et al., 2018. However, these approaches may not capture the complexity of planning decisions or predicting other drivers' behavior, and their underlying supervised learning approach could be unable to cope with multiple complex sub-problems at once, including sub-problems not relevant to the driving task itself Sallab et al., 2017. There are thus many reasons to consider a RL self-driving framework, which can tackle driving problems by interacting with an environment and learning from experience Sallab et al., 2017.

An example of an autonomous driving task implementation, based on Inverse Reinforcement Learning (IRL), was proposed by Sharifzadeh et al. (2016). The authors claimed that, in such a large state space task as driving, IRL can be effective in extracting the reward signal, using driving data from experts demonstrations. End-to-end low-level control through a RL driver was done by Jaritz et al. (2018), in a simulated environment, based on the racing game TORCS, in which the driver has to learn full control of its car, that is steering, brake, gas, and even hand brake to enforce drifting. Autonomous driving is a challenging task for RL because it needs to ensure functional safety and every driver has to deal with the potentially unpredictable behavior of others Shalev-Shwartz, Shammah, and Shashua, 2016. One of the most interesting aspects of autonomous driving is learning how to efficiently cross an intersection, which requires providing suitable information on the intersection to the RL drivers Qiao et al., 2018, as well as correctly negotiating the access with other non-learning drivers and observing their trajectory Tram et al., 2018; Liebner et al., 2012. Safely accessing to an intersection is a challenging task for RL drivers, due to the nature of the intersection itself, which may be occluded, and possible obstacles might not be clearly visible Isele et al., 2017. Another interesting aspect for RL drivers is learning to overtake other cars, which can be a particularly challenging task, depending on the shape of the road section in which the cars are placed Loiacono et al., 2010, but also depending on the vehicles size, as in Hoel, Wolff, and Laine, 2018, where a RL driver learns to control a truck-trailer vehicle in an highway with other regular cars. The authors of Grigorescu et al., 2019; Kiran et al., 2020 provided extensive classifications of the AI state-of-the-art techniques employed in autonomous driving, together with the degrees of automation that are possible for self-driving cars.

Despite the engineering advancements in designing self-driving cars, a lack of legal framework for these vehicles might slow down their coming Brodsky, 2016. There are also important ethical and social considerations. It has been proposed to address the corresponding issues as an engineering problem, by translating them into algorithms to be handled by the embedded software of a self-driving car Holstein, Dodig-Crnkovic, and Pelliccione, 2018. This way the solution of a moral dilemma should be calculated based on a given set of rules or other mechanisms—although the exact practical details and, most importantly, their corresponding implications, are unclear. The problem of autonomous vehicles regulation is particularly relevant in mixed-traffic scenarios, as stated by Nyholm and Smids (2018) and Kirkpatrick (2015), as human drivers may behave in unpredictable ways to the machines. This problem could be mitigated by providing human drivers with more technological devices to help them drive more similar to robotic drivers, but mixed traffic ethics certainly introduce much deeper and more difficult problems Nyholm and Smids, 2018.

A formalization of traffic rules for autonomous vehicles was provided by Rizaldi and Althoff (2015), according to which a vehicle is not responsible for a collision if satisfying all the rules while colliding. Another driving automation approach

based on mixed traffic rules is proposed in Vanholme et al., 2013, where the rules are inspired by current traffic regulation. Traffic rules synthesis could even be automated, as proposed in Medvet, Bartoli, and Talamini, 2017, where a set of rules is evolved to ensure traffic efficiency and safety. The authors considered rules expressed by means of a language generated from a Backus–Naur Form grammar O'Neill and Ryan, 2001, but other ways to express spatiotemporal properties have been proposed Nenzi et al., 2015; Bartocci et al., 2017. Given the rules, the task of automatically finding the control strategy for robotics systems with safety rules is considered in Tumova et al., 2013, where the agents have to solve the task while minimizing the number of violated rules. AI safety can be inspired by humans, who intervene on agents in order to prevent unsafe situations, and then by training an algorithm to imitate the human intervention Saunders et al., 2018, thus reducing the amount of human labour required. A different strategy is followed by Mirchevska et al., 2018, where the authors defined a custom set of traffic rules based on the environment, the driver, and the road graph. With these rules, a RL driver learns to safely make lane-changing decisions, where the driver's decision making is combined with the formal safety verification of the rules, to ensure that only safe actions are taken by the driver A similar approach is considered in Bouton et al., 2019, where the authors replaced the formal safety verification with a learnable safety belief module, as part of the driver's policy.

4.3 Model

We consider a simple road traffic scenario in the form of a directed graph where the road sections are edges, and the intersections are vertices. Each road element is defined by continuous linear space in the direction of its length, and an integer number of lanes. In this scenario, a fixed number of cars move on the road graph according to their driver decisions for a given number of discrete time steps.

4.3.1 Road Graph

A *road graph* is a directed graph $G = (S, I)$ in which edges E represent road sections, and vertices I represent road intersections. Each road element $p \in G$ is connected to the next elements $n(p) \subset G$, with $n(p) \neq \emptyset$. Edges are straight one-way roads with one or more lanes. For each edge p, it holds that $n(p) \subset I$. Vertices can be either turns or crossroads, have exactly one lane, and are used to connect road sections. For each vertex p it holds that $n(p) \subset S$, and $|n(p)| = 1$. Every road element $p \in G$ is defined by its length $l(p) \in \mathbb{R}^+$, and its number of lanes $w(p) \in \mathbb{N}, w > 0$. We do not take into account traffic lights or roundabouts in this scenario.

4.3.2 Cars

A *car* simulates a real vehicle that moves on the road graph G: its position can be determined at any time of the simulation in terms of the currently occupied road element and current lane. The car movement is determined in terms of two speeds—the *linear speed* along the road element and the *lane-changing speed* along the lanes of the same element. At each time step, the car state is defined by the tuple (p, x, y, v_x, v_y, s), where $p \in \{S, I\}$ is the current road element, $x \in [0, l(p)]$ is the position on the road element, $y \in \{1, \ldots, w(p)\}$ is the current lane, $v_x \in [0, v_{\max}]$ is the linear speed, $v_y \in \{-1, 0, 1\}$ is the lane-changing speed, and $s \in \{\text{alive}, \text{dead}\}$ is

the status (time reference is omitted for brevity). All the cars have the same length l_{car} and the same maximum speed $v_{\max}$.

At the beginning of a simulation, all cars are placed uniformly among the road sections, on all the lanes, ensuring that a minimum distance exists between cars i, j on the same road element $p_i = p_j$, such that: $|x_i - x_j| > x_{\text{gap}}$. The initial speeds for all the cars are $v_x = v_y = 0$, and the status is $s =$ alive.

At the next time steps, if the status of a car is $s =$ dead, the position is not updated. Otherwise, if the status is $s =$ alive, the position of a car is updated as follows. Let $\left(a_x^{(t)}, a_y^{(t)}\right) \in \{-1, 0, 1\} \times \{-1, 0, 1\}$ be the driver action composed, respectively, of $a_x^{(t)}$ accelerating action and $a_y^{(t)}$ lane-changing action (see details below). The linear speed and the lane-changing speed at time $t+1$ are updated accordingly with the driver action $\left(a_x^{(t)}, a_y^{(t)}\right)$ at time t as:

$$v_x^{(t+1)} = \min\left(v_{\max}, \max\left(v_x^{(t)} + a_x^{(t)} a_{\max} \Delta t, 0\right)\right) \tag{4.1}$$

$$v_y^{(t+1)} = a_y^{(t)} \tag{4.2}$$

where $a_{\max}$ is the intensity of the *instant acceleration* and Δt is the discrete time step duration. The car linear position on the road graph at time $t+1$ is updated as:

$$x^{(t+1)} = \begin{cases} x^{(t)} + v_x^{(t+1)} \Delta t & \text{if } v_x^{(t+1)} \Delta t \leq x_{\text{stop}}^{(t)} \\ v_x^{(t+1)} \Delta t - x_{\text{stop}}^{(t)} & \text{otherwise} \end{cases} \tag{4.3}$$

where x_{stop} is the distance ahead to the next road element, and is computed as:

$$x_{\text{stop}}^{(t+1)} = l\left(p^{(t+1)}\right) - x^{(t+1)} \tag{4.4}$$

The car lane position at time $t+1$ is updated as:

$$y^{(t+1)} = \min\left(w(p^{(t+1)}), \max\left(y^{(t)} + v_y^{(t+1)}, 1\right)\right) \tag{4.5}$$

The road element at time $t+1$ is computed as:

$$p^{(t+1)} = \begin{cases} p^{(t)} & \text{if } v_x^{(t)} \Delta t \leq x_{\text{stop}}^{(t)} \\ \sim U\left(n\left(p^{(t)}\right)\right) & \text{otherwise} \end{cases} \tag{4.6}$$

where U is the uniform distribution over the next road elements coming from p In other words, when exiting from an intersection, a car enters an intersection chosen randomly from $n\left(p^{(t)}\right)$.

Two cars collide, if the distance between them is smaller than the cars length l_{car}. In particular, for any cars (p, x, y, v_x, v_y, s), $(p', x', y', v_x', v_y', s')$, the status at time $t+1$ is updated as (we omit the time superscript for readability):

$$s = \begin{cases} \text{dead} & \text{if } \left(p = p' \wedge |x - x'| < l_{\text{car}}\right) \vee \left(p' \in n(p) \wedge x_{\text{stop}} + x' < l_{\text{car}}\right) \\ \text{alive} & \text{otherwise} \end{cases} \tag{4.7}$$

When a collision occurs, we simulate an impact by giving the leading car a positive acceleration of intensity a_{coll}, while giving the following car a negative acceleration of intensity $-a_{\text{coll}}$, for the next t_{coll} time steps. Collided cars are kept in the simulation

for the next $t_{\text{dead}} > t_{\text{coll}}$ time steps of the simulation, thus acting as obstacles for the alive ones.

4.3.3 Drivers

A *driver* is an algorithm that is associated to a car. Each driver is able to sense part of its car variables and information from the road environment, and takes driving actions that affect its car state. Every driver ability to see obstacles on the road graph is limited to the distance of view d_{view}.

Observation

For the driver of a car (p, x, y, v_x, v_y, s), the set of visible cars in the jth relative lane, with $j \in \{-1, 0, 1\}$, is the union of the set $V_{\text{same},j}$ of cars that are in the same segment and the same or adjacent lane and the set V_{next} of cars that are in one of the next segments $p' \in n(p)$, in both cases with a distance shorter than d_{view}:

$$V_{\text{same},j} = \left\{(p', x', y', v'_x, v'_y, s') : p' = p \wedge 0 < x' - x \leq d_{\text{view}} \wedge y' = y + j\right\} \tag{4.8}$$

$$V_{\text{next}} = \left\{(p', x', y', v'_x, v'_y, s') : p' \in n(p) \wedge x_{\text{stop}} + x' \leq d_{\text{view}}\right\} \tag{4.9}$$

We remark that the set of cars $V_j = V_{\text{same},j} \cup V_{\text{next}}$ includes also the cars in the next segments: the current car is hence able to perceive cars in a intersection, when in a segment, or in the connected sections, when in an intersection, provided that they are closer than d_{view}.

The driver's observation is based on the concept of *jth lane closest car* c_j^{closest}, based on the set V_j defined above. For each driver, c_j^{closest} is the closest one in V_j:

$$c_j^{\text{closest}} = \begin{cases} \underset{(p',x',y',v'_x,v'_y,s') \in V_j}{\arg\min} \quad \mathbb{1}(p' = p)(x' - x) + \mathbb{1}(p' \neq p)(x_{\text{stop}} + x') & \text{if } V_j \neq \varnothing \\ \varnothing & \text{otherwise} \end{cases} \tag{4.10}$$

where $V_j = V_{\text{same},j} \cup V_{\text{next}}$ and $\mathbb{1} : \{\text{false}, \text{true}\} \to \{0, 1\}$ is the indicator function. Figure 4.1 illustrates two different examples of jth lane closest car, with $j = 0$. We can see that the c_j^{closest} might not exist for some j, either if there is no car closer than d_{view} or if there is no such jth lane.

d

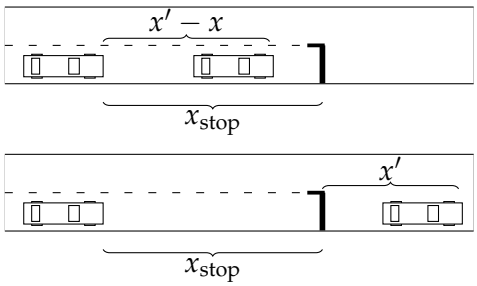


FIGURE 4.1: Distance between cars in different cases.

We define the *closeness* variables $\delta_{x,j} \in [0, d_{\text{view}}]$, with $j \in \{-1, 0, 1\}$, as the distances to the jth lane closest cars c_j^{closest}, if any, or d_{view}, otherwise. Similarly, we

define the *relative speed* variables $\delta_{v,j} \in [-v_{\max}, v_{\max}]$, with $j \in \{-1, 0, 1\}$, as the speed difference of the current car with respect to the jth lane closest cars c_j^{closest}, if any, or $v_{\max}$, otherwise.

At each time step of the simulation, each driver observes the distance from its car to the next road element, indicated by x_{stop}, the current lane y, the current linear speed v_x, the status of its vehicle s, the road element type $e = \mathbb{1}(p \in S)$ its car is currently on, the closeness variables $\delta_{x,j}$, and the relative speed variable $\delta_{v,j}$. We define each driver *observation* as: $o = (x_{\text{stop}}, y, v_x, s, e, \delta_{x,-1}, \delta_{x,0}, \delta_{x,1}, \delta_{v,-1}, \delta_{v,0}, \delta_{v,1})$, therefore $o \in O = [0, l_{\max}] \times \{1, w_{\max}\} \times [0, v_{\max}] \times \{\text{alive}, \text{dead}\} \times \{0, 1\} \times [0, d_{\text{view}}]^3 \times [-v_{\max}, v_{\max}]^3$.

Action

Each agent action is $a = (a_x, x_y) \in A = \{-1, 0, 1\} \times \{-1, 0, 1\}$. Intuitively a_x is responsible for updating the linear speed in the following way: $a_x = 1$ corresponds to accelerating, $a_x = -1$ corresponds to breaking, and $a_x = 0$ keeps the linear speed unchanged. On the other hand a_y is responsible for updating the lane-position in the following way: $a_y = 1$ corresponds to moving to the left lane, $a_y = -1$ corresponds to moving to the right lane, and $a_y = 0$ to keeping the lane-position unchanged.

4.3.4 Rules

A *traffic rule* is a tuple (b, w) where $b : O \to \{false, true\}$ is the rule predicate, defined on the drivers observation space O, and $w \in \mathbb{R}$ is the rule weighting factor. The ith driver *breaks* a rule at a given time step t if the statement b that defines the rule is $b(o_i^{(t)}) = 1$. We define a set of three rules $((b_1, w_1), (b_2, w_2), (b_3, w_3))$, described in the next sections, that we use to simulate the real-world traffic rules for the drivers. All the drivers are subjected to the rules.

Intersection Rule

In this road scenario, we do not enforce any junction access negotiation protocol, nor we consider traffic lights, and cars access interactions as in Figure 4.2. That is, there is no explicit reason for drivers to slow down when approaching a junction, other than the chances of collisions with other cars crossing the intersection at the same time. Motivated by this lack of safety at intersections, we define a traffic rule that punishes drivers approaching or crossing an intersection at high linear speed.

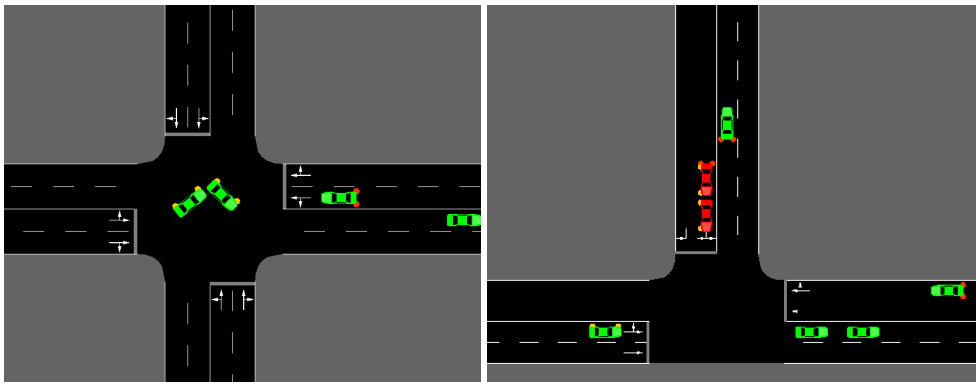


FIGURE 4.2: Cars approaching intersections.

In particular, the driver in road element p such that $p \in I$ is an intersection, or equivalently $p \in S$ and its car is in the proximity of an intersection, denoted by $x_{\text{stop}} < 2l_{\text{car}}$, breaks the intersection rule indicated by (b_1, w_1) if traveling at linear speed $v_x > 10$:

$$b_1(o) = \begin{cases} 1 & \text{if } \left(p \in I \vee x_{\text{stop}} < 2l_{\text{car}}\right) \wedge v_x > 10 \\ 0 & \text{otherwise} \end{cases} \tag{4.11}$$

Distance Rule

Collisions may occur when traveling with insufficient distance from the car ahead, since it is difficult to predict the leading car behavior in advance. For this reason, we introduce a rule that punishes drivers that travel too close to the car ahead.

In particular, the driver observing c_0^{closest} closest car on the same lane breaks the distance rule indicated by (b_2, w_2) if traveling at linear speed v_x such that the distance traveled before arresting the vehicle is greater than $\delta_{x,0} - l_{\text{car}}$, or, in other words:

$$b_2(o) = \begin{cases} 1 & \text{if } \delta_{x,0} - l_{\text{car}} < 2a_{\max} v_x^2 \\ 0 & \text{otherwise} \end{cases} \tag{4.12}$$

Right Lane Rule

In this scenario, cars might occupy any lane on a road segment, without any specific constraint. This freedom might cause the drivers to unpredictably change lanes while traveling, thus endangering other drivers, who might not have the chance to avoid the oncoming collision. Motivated by this potentially dangerous behaviors, we define a rule that allows drivers to overtake when close to the car ahead, but punishes the ones leaving the right-most free lane on a road section.

In particular, the driver occupying road section $p \in S$, on non-rightmost lane $y > 1$, breaks the right lane rule indicated by (b_3, w_3) if the closest car on the right lane c_{-1}^{closest} is traveling at a distance $\delta_{x,-1} = d_{\text{view}}$:

$$b_3(o) = \begin{cases} 1 & \text{if } p \in S \wedge y > 1 \wedge \delta_{x,-1} = d_{\text{view}} \\ 0 & \text{otherwise} \end{cases} \tag{4.13}$$

4.3.5 Reward

Drivers are rewarded according to their *linear speed*, thus promoting efficiency. All cars involved in a collision, denoted by state $s = \text{dead}$, are then arrested after the impact, thus resulting in zero reward for the next $t_{\text{dead}} - t_{\text{coll}}$ time steps, which implicitly promotes safety. Each driver reward at time t is:

$$r^{(t)} = \frac{v_x^{(t)}}{v_{\max}} - \sum_{i=1}^{3} w_i b_i(o^{(t)}) \tag{4.14}$$

where w are the weights of the rules.

4.3.6 Policy Learning

Each driver's goal is to maximize the *return* over a simulation, indicated by $\sum_{t=0}^{T} \gamma^t r^{(t+1)}$, where $\gamma \in [0, 1]$ is the discount factor and $T > 0$ is the number of time steps of the

simulation. The driver *policy* is the function $\pi_\theta : O \to A$ that maps observations to actions. We parameterize the drivers' policy in the form of a feed-forward neural network, where θ is the set of parameters of the neural network. Learning the optimal policy corresponds to the problem of finding the values of θ that maximize the return over an entire simulation. We perform policy learning by means of RL.

4.4 Experiments

Our goal was to experimentally assess the impact of the traffic rules on the optimized policies, in terms of overall efficiency and safety. To this aim, we defined 3 tuples, which are, respectively, the reward tuple R, the efficiency tuple E, and the collision tuple C.

The *reward* tuple $R \in \mathbb{R}^{n_{\text{car}}}$ is the tuple of individual rewards collected by the drivers during an episode, from $t = 0$ to $t = T$, and is defined as:

$$R = \left(\sum_{t=0}^{T} r_1^{(t)}, \ldots, \sum_{t=0}^{T} r_{n_{\text{cars}}}^{(t)} \right) \tag{4.15}$$

The *efficiency* tuple $E \in \mathbb{R}^{n_{\text{car}}}$ is the tuple of sums of individual instant *linear speed* v_x for each driver during an episode, from $t = 0$ to $t = T$, and is defined as:

$$E = \left(\sum_{t=0}^{T} v_{x_1}^{(t)}, \ldots, \sum_{t=0}^{T} v_{x_{n_{\text{cars}}}}^{(t)} \right) \tag{4.16}$$

The *collision* tuple $C \in \mathbb{N}^{n_{\text{car}}}$ is the tuple of individual collisions for each driver during an episode, from $t = 0$ to $t = T$, and is defined as:

$$C = \left(\sum_{t=0}^{T} \mathbb{1}\{s_1^{(t-1)} = \text{alive} \wedge s_1^{(t)} = \text{dead}\}, \ldots, \sum_{t=0}^{T} \mathbb{1}\{s_{n_{cars}}^{(t-1)} = \text{alive} \wedge s_{n_{cars}}^{(t)} = \text{dead}\} \right) \tag{4.17}$$

Each ith element c_i of this tuple is defined as the number of times in which the ith driver change its car status s_i from $s_i = \text{alive}$ to $s_i = \text{dead}$ between 2 consecutive time steps $t - 1$ and t.

We considered 2 different driving scenarios in which we aimed at finding optimal policy parameters:y "no-rules", in which traffic rules weighting factors are $w_1 = w_1 = w_3 = 0$, such that drivers are not punished for breaking the rules, and "rules", in which traffic rules weighting factors are $w_1 = w_2 = w_3 = 1$, such that drivers are punished for breaking the rules, and all the rules have the same relevance.

Moreover, we considered 2 different collision scenarios:

(a) cars are kept with status $s = \text{dead}$ in the road graph for t_{dead} time steps, and then are removed; and

(b) cars are kept with status $s = \text{dead}$ in the road graph for t_{dead} time steps, and then their status is changed back into $s = \text{alive}$.

The rationale for considering the second option is that the condition in which we remove collided cars after t_{dead} time steps may not be good enough for finding the optimal policy. This assumption could ease the task of driving for the non-collided cars, when the number of collided cars grows, and, on the other side, it might provide too few collisions to learn from.

We simulated n_{cars} cars sharing the same driver policy parameters and moving in the simple road graph in Figure 4.3 for T time steps. This road graph has 1 main intersection at the center, and 4 three-way intersections. All road segments $p \in S$ have the same length $l(p)$ and same number of lanes $w(p)$. We used the model parameters shown in Table 4.1 and performed the simulations using Flow Wu et al., 2017, a microscopic discrete-time continuous-space road traffic simulator that allows implementing our scenarios.

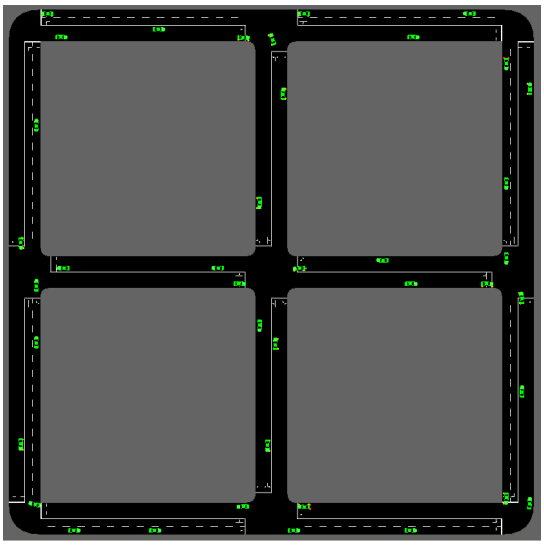


FIGURE 4.3: The road graph used in the experiments.

TABLE 4.1: Model and simulation parameters.

Param	Meaning	Value
l_{car}	Car length	7
t_{coll}	Impact duration	10
t_{dead}	Collision duration	20
d_{view}	Driver's view distance	50
v_{max}	Driver's maximum speed	50
a_{max}	Driver's acceleration (deceleration)	2
Δt	Time step duration	0.2
$\|S\|$	Number of road sections	12
$\|I\|$	Number of road intersections	9
$w(p), p \in G$	Number of lanes	$\in \{1,2\}$
$l(p), p \in S$	Section length	100
n_{car}	Cars in the simulation	40
T	Simulation time steps	500

We repeated n_{trials} experiments in which we performed n_{train} training iterations in order to optimize the initial random policy parameters $\theta_{\text{no-rules}}$ and θ_{rules}. We collected the values, across the n_{trials} repetitions, of R, E, and C during the training.

We employed Proximal Policy Optimization (PPO) Schulman et al., 2017 as the RL policy optimization algorithm: PPO is a state-of-the-art actor-critic algorithm that is highly effective, while being almost parameters-free. We used the PPO default configuration (`https://ray.readthedocs.io/en/latest/rllib-algorithms.html`) with the parameters shown in Table 4.2. The drivers policy is in the form of an

actor-critic neural networks model, where each of the 2 neural networks is made of 2 hidden layers, each one with 256 neurons and hyperbolic tangent as activation function. The hidden layer parameters are shared between the actor and the critic networks: this is a common practice introduced by Mnih et al. (2016) that helps to improve the overall performances of the model. The parameters of the actor network as well as the ones of the critic network are initially distributed according to the *Xavier initializer* Glorot and Bengio, 2010.

TABLE 4.2: Policy learning algorithm parameters.

Param	Meaning	Value
n_{trial}	Number of trials	20
n_{train}	Training iterations	500
n_{car}	Cars in the simulation	40
γ	Discount factor	0.999

4.5 Results

Figures 4.4 and 4.5 show the training results in terms of the tuples R, E, and C for the 2 policies $\theta_{\text{no-rules}}$ and θ_{rules} in the two collision scenarios considered.

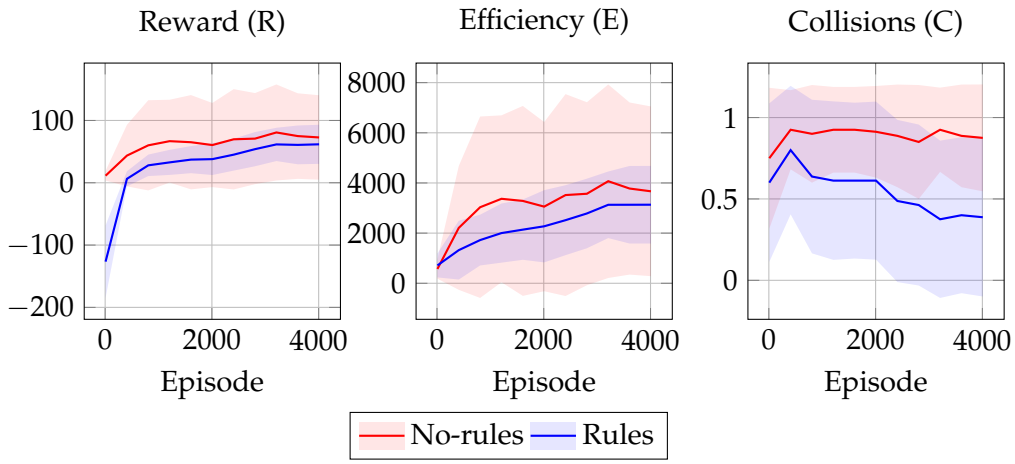


FIGURE 4.4: Training results with cars removed after t_{dead} time steps. Here, we draw the training values of R, E, and C, at a certain training episode, averaged on n_{trial} experiments. We indicate with solid lines the mean of R,E, and C among the n_{car} vehicles, and with shaded areas their standard deviation among the n_{car} vehicles.

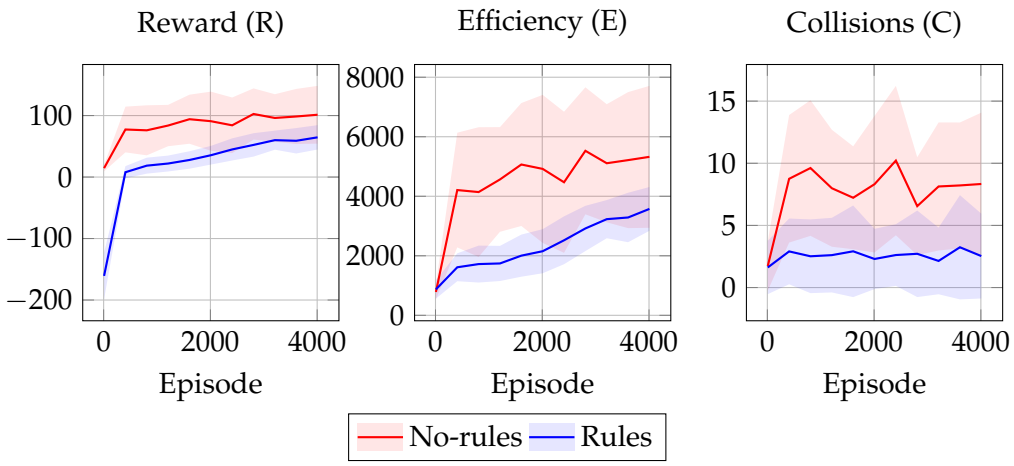


FIGURE 4.5: Training results with cars restored after t_{dead} time steps. Here, we draw the training values of R, E, and C, at a certain training episode, averaged on n_{trial} experiments. We indicate with solid lines the mean of R,E, and C among the n_{car} vehicles, and with shaded areas their standard deviation among the n_{car} vehicles.

In all experimental scenarios, the policy learned with rules shows driving behaviors that are less efficient than the ones achieved by the one without rules. On the other hand, the policy learned without rules is not even as efficient as it could theoretically be, due to the high number of collisions that make it difficult to avoid collided cars. Moreover, the values of E for the drivers employing the rules are distributed closer to the mean efficiency value, and thus we can assume this is due to the fact that the rules limit the space of possible behaviors to a smaller space with respect to the case without rules. In other words, rules seems to favor equity among drivers.

On the other hand, the policy learned with rules shows driving behaviors that are safer than the ones achieved by the one without rules. This may be due to the fact that training every single driver to avoid collisions based only on the efficiency reward is a difficult learning task, as well as because agents are not capable of predicting the other agents' trajectories. On the other hand, we can see that the simple traffic rules that we have designed are effective at improving the overall safety.

In other words, these results show that, as expected, policies learned with rules are safer but less efficient than the ones without rules. Interestingly, the rules act also as a proxy for equality, as shown in Figures 4.4 and 4.5, in particular for the efficiency values of E, where the blue shaded area is much thinner than the red one, meaning that all the n_{car} vehicles have similar efficiency.

4.5.1 Robustness to traffic level

With the aim of investigating the impact of the traffic level on the behavior observed with the learned policies (in the second learning scenario), we performed several other simulations by varying the number of cars in the road graph. Upon each simulation, we measured the overall distance traveled $\sum_{i=1}^{n_{\text{car}}} E_i \Delta t$ and overall collisions $\sum_{i=1}^{n_{\text{car}}} C_i$. We considered the overall sums, instead of the average, of these indexes in order to investigate the impact of the variable number of cars in the graph: in principle, the larger is this number, the longer is the overall distance that can be potentially traveled, and, likely, the larger is the number of collisions.

We show the results of this experiment in Figure 4.6, where each point corresponds to indexes observed in a simulation with a given traffic level n_{car}: we considered values in $10, 20, \ldots, 80$. We repeated the same procedure for both the drivers trained with and without the rules, using the same road graph in which the drivers have been trained. For each level of traffic injected, we simulated T time steps and we measured the overall distance and overall number of collisions occurred.

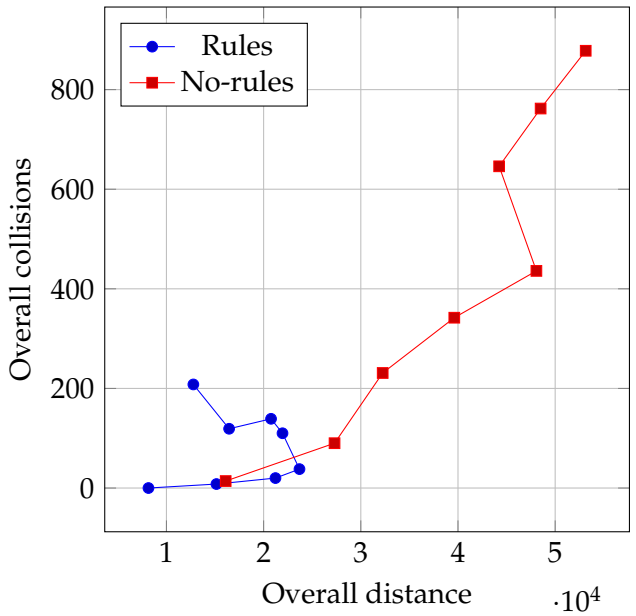


FIGURE 4.6: Overall number of collisions in the simulation against the overall traveled distance in the simulation, averaged across simulations with the same n_{car}. Each dot is drawn from the sum of the values computed on the n_{car} vehicles.

As shown in Figure 4.6, the two policies (corresponding to learning with and without rules) exhibit very different outcomes as the injected traffic increases. In particular, the policy optimized without rules results in an overall number of collisions that increases, apparently without any bound in these experiments, as the traffic level increases. Conversely, the policy learned with the rules keeps the overall number of collisions much lower also with heavy traffic. Interestingly, the limited increase in collisions is obtained by the policy with the rules at the expense of overall traveled distance, i.e., of traveling capacity of the traffic system.

From another point of view, Figure 4.6 shows that a traffic system where drivers learned to comply with the rules is subjected to congestion: when the traffic level exceeds a given threshold, introducing more cars in the system does not allow obtaining a longer traveled distance. Congestion is instead not visible (at least not in the range of traffic levels that we experimented with) with policies learned without rules; the resulting system, however, is unsafe. Overall, congestion acts here as a mechanism, induced by rules applied during the learning, for improving the safety of the traffic system.

4.6 Conclusions

We investigated the impact of imposing traffic rules while learning the policy for AI-powered drivers in a simulated road traffic system. To this aim, we designed a road

traffic model that allows analyzing system-wide properties, such as efficiency and safety, and, at the same time, permits learning using a state-of-the-art RL algorithm.

We considered a set of rules inspired by real traffic rules and performed the learning with a positive reward for traveled distance and a negative reward that punishes driving behaviors that are not compliant with the rules. We performed a number of experiments and compared them with the case where rules compliance does not impact on the reward function.

The experimental results show that imposing the rules during learning results in learned policies that gives safer traffic. The increase in safety is obtained at the expense of efficiency, i.e., drivers travel, on average, slower. Interestingly, the safety is also improved after the learning—i.e., when no reward exists, either positive or negative—and despite the fact that, while training, rules are not enforced. The flexible way in which rules are taken into account is relevant because it allows the drivers to learn whether to evade a certain rule or not, depending on the current situation, and no action is prohibited by design: rules stand hence as guidelines, rather then obligation, for the drivers. For instance, a driver might have to overtake another vehicle in a situation in which overtaking is punished by the rules, if this decision is the only one that allows avoiding a forthcoming collision.

Our work can be extended in many ways. One theme of investigation is the robustness of policies learned with rules in the presence of other drivers, either AI-driven or human, who are not subjected to rules or perform risky actions. It would be interesting to assess how the driving policies learned with the approach presented in this study operate in such situations.

From a broader point of view, our findings may be useful in the situations where there is a trade-off between compliance with the rules and a greater good. With the ever increasing pervasiveness of AI-driven automation in many domains (e.g., robotics and content generation), relevance and quantity of these kinds of situations will increase.

Chapter 5

Sensing Controllers for Voxel-based Soft Robots

Soft robots allow for interesting morphological and behavioral designs because they exhibit more degrees of freedom than robots composed of rigid parts. In particular, *voxel-based soft robots (VSRs)*—aggregations of elastic cubic building blocks—have attracted the interest of Robotics and Artificial Life researchers. VSRs can be controlled by changing the volume of individual blocks: simple, yet effective controllers that do not exploit the feedback of the environment, have been automatically designed by means of Evolutionary Algorithms (EAs).

In this work we explore the possibility of evolving *sensing controllers* in the form of artificial neural networks: we hence allow the robot to sense the environment in which it moves. Although the search space for a sensing controller is larger than its non-sensing counterpart, we show that effective sensing controllers can be evolved which realize interesting locomotion behaviors. We also experimentally investigate the impact of the VSR morphology on the effectiveness of the search and verify that the sensing controllers are indeed able to exploit their sensing ability for better solving the locomotion task.

5.1 Introduction

Traditionally, robots have been made using rigid parts connected by joints. This allowed engineers to model robots behaviour and eased the design of body and controllers for the robots. On the other hand, creatures in nature are composed also, or mainly, of soft tissues and are quite effective in solving many complex tasks which are still utterly hard for robots (Kim, Laschi, and Trimmer, 2013). Inspired by nature (Lin, Leisk, and Trimmer, 2011), in the recent years many researchers focused on robots made on soft tissues, called *soft robots* (Rus and Tolley, 2015). The efforts concerned methods for the assisted or automated design of soft robot bodies (Cheney et al., 2013; Cheney, Clune, and Lipson, 2014) and controllers (Braganza et al., 2007; Vaughan, 2018), often by means of simulation, and techniques for building actual soft robots (Iida and Laschi, 2011; Shepherd et al., 2011).

Voxel-based Soft Robots (VSRs) are a particular category of soft robots. They are aggregations of small elastic cubic building blocks called *voxels* (Hiller and Lipson, 2012). VSRs have been important for the raise of the *embodied cognition paradigm* according to which the complexity of behavior of a (virtual) creature depends on both its brain and its body (Pfeifer and Bongard, 2006). According to this paradigm, a robot should be designed by considering brain and body together rather than by focusing only on its brain, i.e., on its controller. This research path has been particularly significant for VSRs, within a common framework in which the ability of the VSR

to interact with the environment derived mainly from its body (Cheney et al., 2013; Cheney, Clune, and Lipson, 2014).

In this paper, we explore the possibility of automatically synthesizing *sensing* controllers for simple VSRs, i.e., controllers which can sense the environment and exploit the gathered information for guiding the robot movements. We consider VSRs in which the sensing is distributed across the full body, i.e., on each voxel composing the VSR. In other words, we consider a VSR as an aggregation of simple parts that can be used both as actuators and as sensors.

We consider three different VSRs, i.e., with different bodies, and synthesize the corresponding controllers for solving a locomotion task. For each VSR we evolved a sensing controller and a more traditional, non-sensing controller. We represent sensing controllers as artificial neural networks (ANNs) whose topology is determined by the body of the robot, while for non-sensing controller we use a simpler representation which has already been already successfully adopted (Kriegman, Cheney, and Bongard, 2018). We synthesize both kinds of controllers with the same EA where, as we will show, the sensing controller corresponds to a larger search space than the non-sensing one, having more parameters. We evolved each VSR in two different environments, i.e, an even surface and an uneven surface.

Our experimental results, obtained by simulation, show that sensing controllers are always more effective than non-sensing ones, regardless of the body of the VSR and of the environment in which they evolved. Moreover, we also find that sensing controllers exhibit behaviors that are more heterogeneous than those of their non-sensing counterparts. Most importantly, we also assess the behavior of controllers in environments *different* from those in which they were evolved and found that sensing controllers are more effective even in such scenarios. This result suggests that sensing controllers are indeed able to exploit their peculiar ability to sense the environment in which they are immersed.

5.2 Related work

The idea of evolving the body and the controller of simulated creatures dates back to '90s (Sims, 1994). In the cited work, the creatures body is modular, and the controller, in the form of an ANN, is distributed among their body components, capable of sensing the environment.

Other attempts to optimize ANNs controlling soft robots have been done later. For instance, Braganza et al., 2007 consider a tentacle-like manipulator which is controlled by an ANN, since the design of a traditional closed-loop controller for this specific robot was considered unfeasible. Another example is the optimization of a locomotion controller in the form of an ANN for a quadruped simulated creature (Vaughan, 2018).

On the other hand, control strategies different than ANNs have led to interesting results. Bruder et al., 2019, for example, have recently designed a linear dynamical model for controlling soft robots, based on a data-driven model: the authors claim that the proposed method, being more traditional and control-oriented, avoids issues of the ANNs acting as the black-boxes.

We remark that the works cited above face the problem of sensing, but are not based on VSRs. Research on VSRs focused more on how to design (often by means of evolutionary computation) the body of the robot: when the controller was of a non-trivial complexity, it had no sensing ability. Nevertheless, interesting behaviors have been found.

First attempts of morphological optimization of VSRs were done by Hiller and Lipson, 2012 and, later, by Cheney et al., 2013. In the latter work, the novelty was mainly in the representation of the morphology and in the corresponding EA, both achieved with CPPN-NEAT (Neuroevolution of Augmented Topologies applied to Compositional Pattern-Producing Networks, Stanley, 2007): because of their ability to compactly describe patterns with repetitions and symmetries (which resemble nature), CPPN proved to be useful for evolving effective VSR morphologies. In that case, the task was locomotion and the controller was actually determined by the morphology, since different materials statically corresponded to different actuations. A similar approach has been applied later by Cheney, Bongard, and Lipson, 2015 for evolving VSRs able to escape from a tight space.

A different kind of control of the VSR, but still not able to sense the environment, has been studied by Cheney, Clune, and Lipson, 2014. The authors proposed to define materials for the voxels in terms of their ability to propagate and react to an activation signal, inspired by properties of real, biological tissues. Morphologies were then evolved with CPPN-NEAT for the locomotion task.

Materials composing VSRs, in particular soft vs. stiff ones, are also the focus of (Bongard et al., 2016). The authors implemented a distributed growth mechanism, in place of actuation by oscillating global signals. The development of VSRs is allowed during their entire life span, acting at a lower time scale than the oscillation. The task is inspired by plants, and consists in growing towards static (possibly multiple) source of light in the environment, thus allowing the VSRs the ability to sense to a certain extent.

VSRs have been used as a case of study also for reasoning about the evolution in different environment (Corucci et al., 2018). The authors of the cited work evolved morphologies on a land environment in comparison with the ones in a water environment. Subsequently, they investigated the effects of an environmental transition, from land to water and the opposite, during the evolution, and they try to explain morphological results. To some degree, we too experiment with VSRs facing different environment: we assess their ability to move in environments which were not seen during the evolution and we show that sensing is beneficial in this scenario.

5.3 Scenario: controlling VSRs

5.3.1 Voxel-based soft robots (VSRs)

A *voxel-based soft robot* (VSR) is an assembly of one or more *voxels*, i.e., cubic building blocks, each linked to up to 6 neighbour voxels. Voxels are also elastic in the sense that their volume may either contract or expand with respect to the resting volume; the volume of each voxel may vary independently of the volume of any other voxel. We consider VSRs composed of a predefined number of voxels n. The *morphology* of a VSR is the way in which its voxels are linked.

We assume a discrete-time physics model in which scale values are set at regular intervals $t = k\Delta t, k \in \mathbb{N}$, where Δt is a parameter.

At any time, each voxel is defined by $s, \boldsymbol{x}, \boldsymbol{v}, \boldsymbol{v}'$, where: s is the *scale*, i.e., the ratio between the current and resting volume of the voxel; $\boldsymbol{x}$, $\boldsymbol{v}$, and $\boldsymbol{v}'$ are the position, velocity, and acceleration of its center.

The behavior of the robot can be determined by imposing a value for the scale of each of its voxels. By varying the scale for each voxel over time, the corresponding positions, velocities and accelerations will vary over time as well depending on how voxels are linked together. In this work we use the physics model presented by

Kriegman et al., 2017. The behavior of the VSR derives hence from the positions, velocities, and acceleration of its composing voxels, which themselves derive from the values imposed to the scale. We call *controller* of the VSR the way in which scale values are set over the time.

In general, a controller may set the values of the scale over the time basing on external input related to the interactions of the VSR with the environment; or, it may set the scale regardless of those interactions. We call the two approaches *sensing* and *non-sensing* controllers, respectively. In the next sections we describe the two specific controllers that we consider in this work.

5.3.2 Non-sensing controller

We consider the simple non-sensing controller proposed by Kriegman, Cheney, and Bongard, 2018 in which the scale s_i of the i-th voxel varies over time according to a sinusoidal signal, which determines the relative scale with respect to a resting value:

$$s_i(k) = s_i^0 + a \sin(2\pi f k \Delta t + \phi_i) \tag{5.1}$$

Frequency f and amplitude a are predefined and identical for all the voxels. Phase ϕ_i and resting value s_i^0 are instead defined separately for each voxel and constitute the parameters $\theta_{\text{NS}} = (s_1^0, \phi_1, \ldots, s_n^0, \phi_n)$ of the controller. It can be seen, hence, that the number of parameters of this non-sensing controller, and therefore the size of the space of the corresponding controller instances, grows linearly with the number n of voxels in the VSR, i.e., $|\theta_{\text{NS}}| = 2n \sim O(n)$.

5.3.3 Sensing controller

We consider a sensing controller in which the VSR senses the environment in terms of the actual scale, velocity, and acceleration of each of its voxels: since these figures are determined also by how the VSR interacts with the environment, e.g., by pushing on the floor, they correspond to sensing the environment. These inputs, along with a single sinusoidal signal $\sin(2\pi f k \Delta t)$, are fed to a feed-forward ANN whose output layer determines the values of the scale to be set for each of the voxels.

More in detail, the ANN is composed of an input layer of $3n + 1$ neurons (the $+1$ being fed with the sinusoidal signal), an hidden layer of h neurons, and an output layer of n neurons. The activation function is the *Rectified Linear Unit* (ReLU). The input layer is fed with the values $s_1(k-1), \|\boldsymbol{v}_1(k-1)\|, \|\boldsymbol{v}_1'(k-1)\|$ of each voxel. Each output neuron emits a value $o_i \in [-1, 1]$ which is then mapped to $[s^0 - \Delta s, s^0 + \Delta s]$, where s_0 and Δs are pre-defined values which are the same for all the voxels. The output of the i-th neuron at time $k\Delta t$ determines the scale $s_i(k)$ of the i-th voxel:

$$s_i(k) = s^0 + \Delta s o_i \tag{5.2}$$

$$\begin{aligned} o_i = \boldsymbol{f}_i(&s_1(k-1), \|\boldsymbol{v}_1(k-1)\|, \|\boldsymbol{v}_1'(k-1)\|, \ldots, \\ &s_n(k-1), \|\boldsymbol{v}_n(k-1)\|, \|\boldsymbol{v}_n'(k-1)\|; \theta_{\text{S}}) \end{aligned} \tag{5.3}$$

where $\|\boldsymbol{v}_i(k-1)\|$ is the norm of the velocity of the i-th voxel at time $(k-1)\Delta t$, $\boldsymbol{f} : \mathbb{R}^{3n+1} \to [0,1]^n$ represents the ANN, and θ_{S} are the ANN parameters (i.e., weights).

Concerning the number of neurons in the hidden layer, we set $h = 0.65n$. It can be seen that the number of parameters of this sensing controller grows with n^2, i.e., $|\theta_{\text{S}}| = 3(n+1)h + hn \sim O(n^2)$.

5.3.4 Instantiating the controller

We instantiate the two controllers, i.e., we determine the values for their parameters θ_{NS} and θ_{S}, by means of evolutionary computation. To this end, we use for both controllers the Evolutionary Algorithm (EA) shown in Algorithm 2, already used by Kriegman, Cheney, and Bongard, 2018 for evolving a non-sensing controller. This EA evolves a fixed size of n_{pop} individuals for n_{gen} generations, each individual being a vector θ of values ($\theta = \theta_{\text{NS}}$ and $\theta = \theta_{\text{S}}$ for the non-sensing and for the sensing controller, respectively). Only a unary genetic operator (mutation) is used: the mutation consists in perturbing each parameter in θ with probability p_{mut}, the amount of perturbation being with a random value randomly sampled from a normal distribution $N(0, \sigma_{\text{mut}})$. When evolving the non-sensing controller, we limit the values of each $s_i^0 \in \theta_{\text{NS}}$ parameter, after the mutation, to the interval $[s^0 - \Delta s, s^0 - \Delta s]$.

The generational model is a $n + m$ with overlapping and individuals are compared using Pareto dominance applied on their fitness and age: the age of the individual is incremented at each generation, whereas new individuals have the age set to 0. In case of tie in a selection (i.e., when one individual has to be selected from a set of individuals on the same Pareto front), individuals with the best fitness are preferred; in case of further tie, the individual is chosen at random. The same criterion is used to determine the *best individual* at the end of the evolution.

```
1  P ← ∅
2  foreach i ∈ {1, ..., n_pop} do
3  |  P ← P ∪ (random(), 0)
4  end
5  foreach i ∈ {1, ..., n_gen} do
6  |  P' ← ∅
7  |  foreach (θ, a) ∈ P do
8  |  |  θ' ← mutate(θ)
9  |  |  P' ← P' ∪ (θ, a + 1)
10 |  |  P' ← P' ∪ (θ', a + 1)
11 |  end
12 |  P' ← P' ∪ (random(), 0)
13 |  P ← select(P', n_pop)
14 end
```

Algorithm 2: The EA for evolving the controller.

The fitness of an individual θ, i.e., a controller for a VSR, measures its ability to perform a given task. In this work, we consider the locomotion task and set the fitness to the distance that the VSR corresponding to the individual travels along the x-axis during a simulation of a predefined amount of n_{sim} time steps. Despite its apparent simplicity, locomotion is considered a benchmark for VSRs (Cheney et al., 2013; Cheney, Clune, and Lipson, 2014; Kriegman, Cheney, and Bongard, 2018).

We remark that other techniques might be used for the purpose of instantiating a controller, given a morphology and a simulator. In particular, for learning the sensing-controller, which is based on ANN, EAs operating on ANNs might be used, e.g., NEAT (Stanley and Miikkulainen, 2002) or CPPN-NEAT (Stanley, 2007). Or, since the considered scenario consists in an autonomous agent that interacts with the environment trying to maximizing a reward (here, the traveled distance), Reinforcement Learning techniques might be used (Duan et al., 2016). However, we leave the exploration of these alternative options to future work, since here we are interested

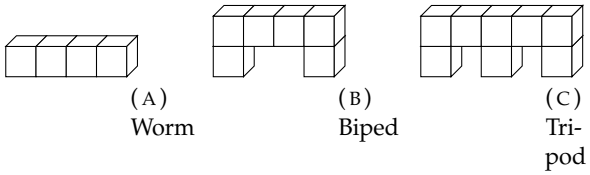


FIGURE 5.1: The three different VSR morphologies.

Param.	Value
Δt	0.14 ms
a	0.001 m^3
f	40 Hz
s^0	0.01 m^3
Δs	0.001 m^3

Param.	Value
n_{pop}	30
n_{gen}	200
p_{mut}	$\frac{1}{\lvert\theta\rvert}$
σ_{mut}	1
n_{sim}	10 000

TABLE 5.1: Parameters of the physics model (top left), morphologies (bottom left), and EA (right) used in the experiments.

in comparing the nature of the controller, and the information it can exploit, rather than the learning technique.

5.4 Experiments and results

We performed an experimental evaluation aimed at investigating the effectiveness of the sensing controller with respect to the non-sensing one. In particular, we aimed at answering the following research questions: (RQ1) Is a sensing controller better than a non-sensing one? (RQ2) Does the larger size of the search space for the sensing controller affect the search effectiveness? (RQ3) Is a sensing controller actually able to exploit its ability to sense the environment? For answering these questions, we considered three different VSR *morphologies* and two different *environments*.

Morphologies are shown in Figure 5.1: we call the corresponding VSRs *worm*, *biped*, and *tripod*. They differ in the number of composing voxels ($n \in \{4, 6, 8\}$) and hence correspond to different numbers of parameters for defining the controllers.

Concerning the environment, we simulated the movement of the VSR on an *even* surface and on an *uneven* surface. In all cases, we performed 30 evolutionary runs (i.e., 30 independent executions of Algorithm 2) for each combination of morphology and environment. We used the implementation made available by Kriegman, Cheney, and Bongard, 2018[1], with the parameters of the physics model, morphologies, and EA shown in Table 5.1. We run the experiments using AWS EC2 on the c4.8xlarge EC2 instances, each equipped with 36 vCPU based on 2.9 GHz Intel Xeon E5-2666 and with 60 G RAM; we distributed the fitness evaluation across the vCPUs and runs across instances.

In each run, the VSR was put in the environment with its main dimension laying on the x-axis, the same axis along which the traveled distance is measured for computing the fitness.

[1] https://github.com/skriegman/how-devo-can-guide-evo

TABLE 5.2: Fitness (in mm, mean μ and standard deviation σ across the 30 runs) of the best individual at the end of the evolution in the environment with even surface. The p-value is computed with the Mann-Whitney U-test (see text).

	Non-sensing		Sensing		p-value
Morph.	μ	σ	μ	σ	$[\times 10^{-3}]$
Worm	146	8	3012	329	0.002
Biped	69	19	931	74	0.006
Tripod	550	26	636	76	0.024

5.4.1 Environment: even surface

Table 5.2 presents the main results obtained in the environment with even surface, with the three morphologies. The table shows the mean μ and the standard deviation σ of the fitness of the best individual at the last generation across the 30 runs. The table also shows the p-values obtained with the Mann-Whitney U-test that we performed for each morphology in order to verify if the samples have the same median.

The foremost finding is that sensing controllers clearly outperform non-sensing ones. That is, a VSR controlled by a sensing controller is in general better in performing the locomotion task, regardless of the morphology. The difference is always statistically significant (with a significance level of $\alpha = 0.05$) and substantial in two on three cases, the worm and the biped.

Concerning the tripod, the sensing controller is still better, in terms of the final best fitness, than the non-sensing one, but the difference is lower (636 ± 76 vs. 550 ± 263) with respect to the worm and biped (for which traveled distance difference is of an order of magnitude). We interpret this finding as a consequence of the fact that the number of voxels in the tripod is larger: the complexity of the controller is $O(n)$ for the non-sensing case and $O(n^2)$ for the sensing case, and the same applies for the size of the search space. As a further evidence for this interpretation, we show in Figure 5.2 how the fitness of the best individual varies during the evolution (mean across the 30 runs) for the three morphologies. Beyond highlighting the lower difference for the tripod, Figure 5.2 suggests that the evolution of a sensing controller has not yet stopped at the end of the evolution (200-th generation), for this case; on the other end, this does not occur with the non-sensing controller. In other words, the larger search space makes finding the optimum harder. We remark, however, that other techniques exist for evolving ANNs which are suitable for scenarios like the one considered in this work. In particular, we argue that NEAT (or its recent variants as, e.g., the one of Silva et al., 2015) might be a way to address the issue of the large search space, thanks to its ability to progressively increase the expressiveness of the representation—i.e., complexification.

Analysis of the behaviors

In order to further investigate the differences between the sensing and non-sensing controllers, we observed the resulting behaviors during the simulations: i.e., we looked at the way best evolved controllers moved and drawn qualitative reasoning (see Figure 5.3). We found that sensing controllers resulted, in general, in a broader set of behaviors, the difference being more apparent for the worm. Interestingly, for

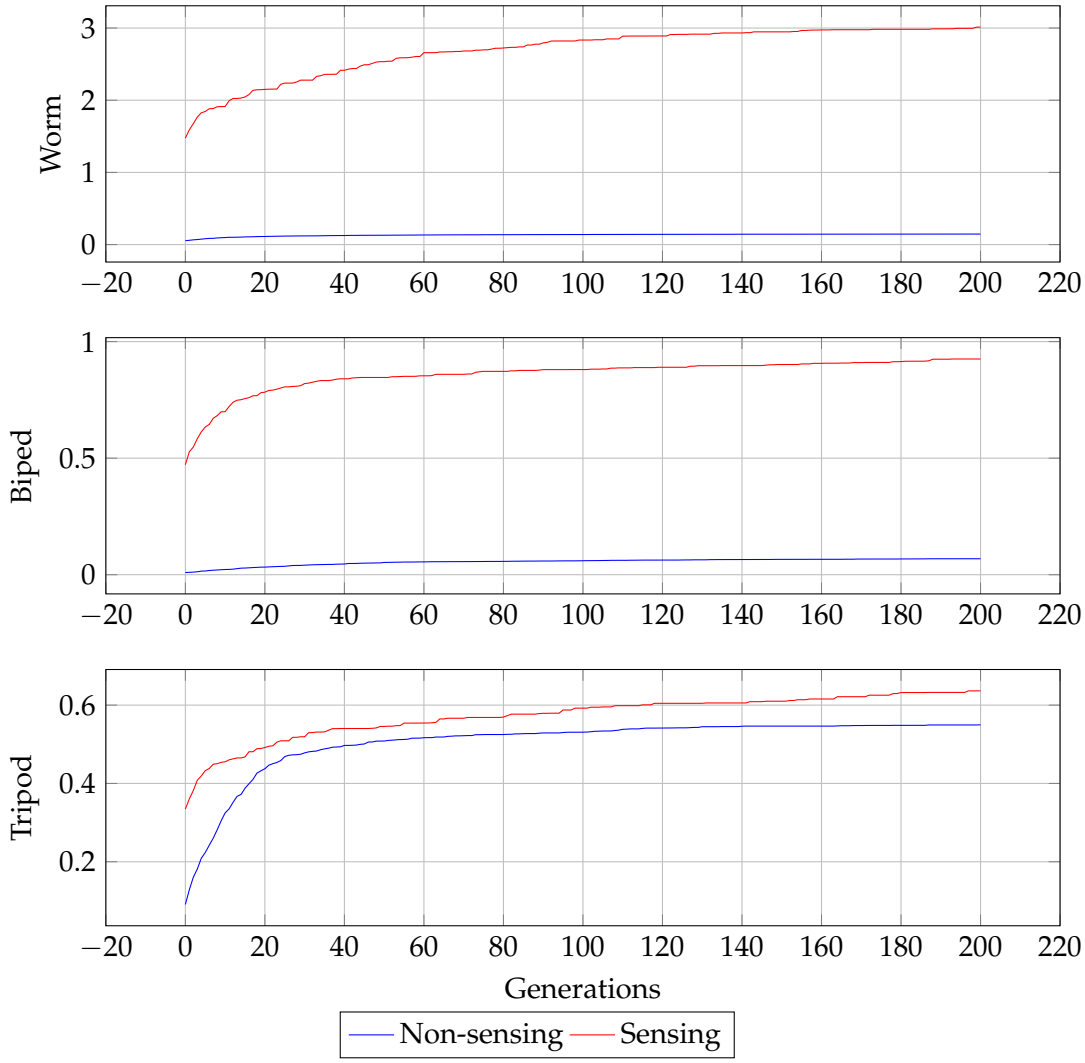


FIGURE 5.2: Fitness (in m, mean across the 30 runs) of the best individual during the evolution.

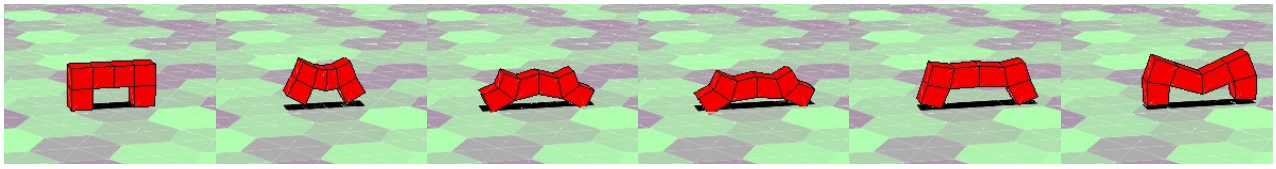


FIGURE 5.3: Frames capturing the behavior of a biped with one of the evolved sensing controller in the environment with even surface.

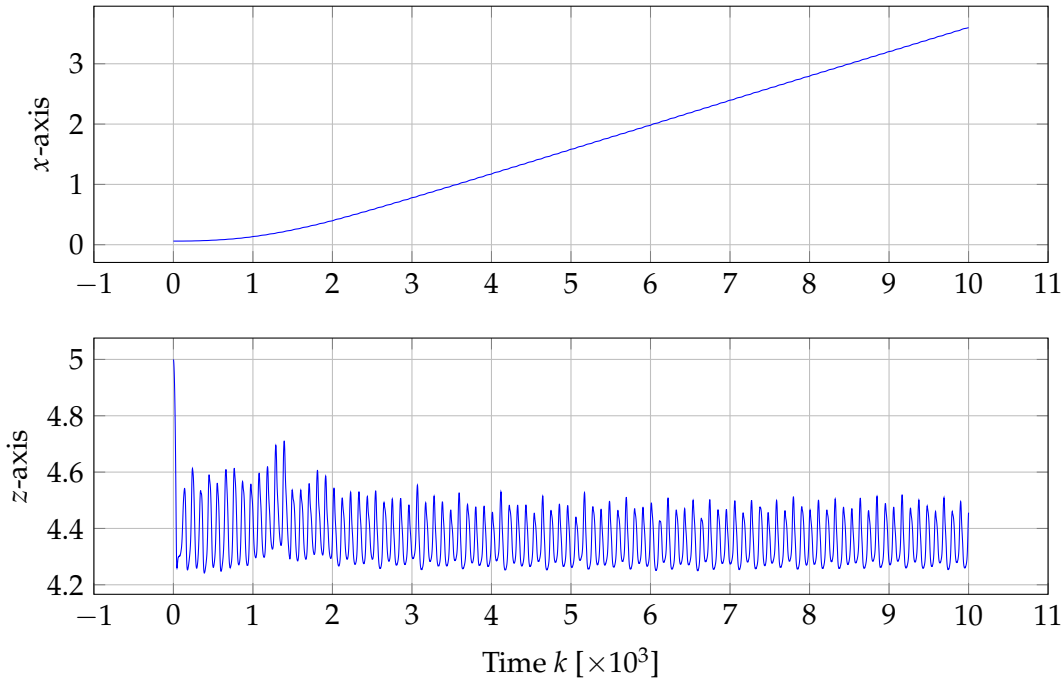


FIGURE 5.4: Trajectory $\boldsymbol{x}_{\text{CM}}(k)$ of the center of mass, shown separately for the two salient axes (scale is $\times 10^{-3}$ for the z-axis), of a worm with one of the evolved sensing controller in the environment with even surface.

this morphology the behaviors exhibited by the sensing controllers often visually resembled those of the real biological counterpart.

In an attempt of quantifying the result of this qualitative analysis, we devised a way of systematically capturing and describing the behaviors of the VSR—similar procedures have been already used for analyzing the behavior of robots with evolved controllers, e.g., in Silva, Correia, and Christensen, 2017. We proceeded as follows. (1) For each morphology, we considered all and only the 60 best controllers (sensing and non-sensing) obtained at the last generation. (2) We considered the discrete signals corresponding to the position $\boldsymbol{x}_{\text{CM}}(k)$ of the center of mass of the VSR during fitness evaluation. Figure 5.4 shows an example trajectory of one of the best sensing controllers for the worm morphology. (3) We computed the discrete Fourier transform (DFT) coefficients $\boldsymbol{d}_x$ and $\boldsymbol{d}_z$, with $\boldsymbol{d}_x, \boldsymbol{d}_z \in \mathbb{R}^{n_{\text{sim}}}$, of the x- and z-components of $\boldsymbol{x}_{\text{CM}}(k)$; we did not consider the y-component since VSRs do not move significantly along that axis (see Figure 5.4). (4) We concatenated $\boldsymbol{d}_x$ and $\boldsymbol{d}_y$, hence obtaining a vector $\boldsymbol{d} \in \mathbb{R}^{2n_{\text{sim}}}$ for each observed behavior. (5) Finally, we mapped all the behaviors from $\mathbb{R}^{2n_{\text{sim}}}$ to $\mathbb{R}^2$ using Multidimensional Scaling (MDS) (Cox and Cox, 2000). We explored different dimensionality reduction techniques, e.g., t-SNE (Maaten and Hinton, 2008): the qualitative observations presented below did not change.

Figure 5.5 shows the results of the analysis of the behaviors: for each morphology,

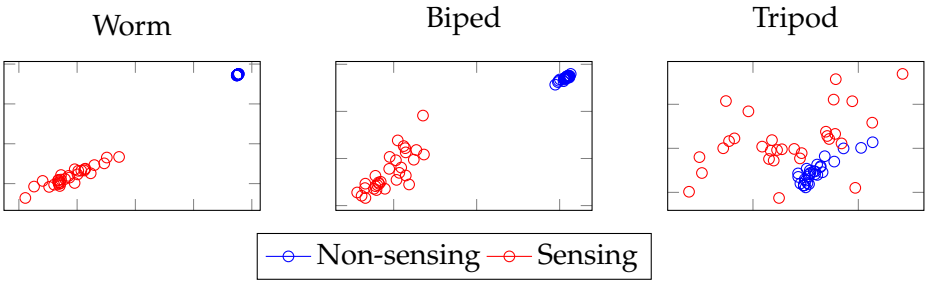


FIGURE 5.5: Behaviors resulting from the 60 best controllers evolved in the environment with even surface, with the three morphologies.

the figure includes a plot where each behavior is a marker positioned according to the first two MDS coordinates. Three observations can be done based Figure 5.5. First, for the two simplest morphologies (worm and biped) the behaviors obtained with sensing and non-sensing controllers look clearly dissimilar: the red cloud is far from the blue cloud. Second, non-sensing controllers result in more homogeneous behaviors than sensing controllers: the red cloud is in general larger than the blue cloud. Third, the tripod case is, consistently with the previous findings, different from the other two cases: the difference of behaviors is fuzzier and similar behaviors can be found which are obtained with different controllers. We think that the motivation for this finding is twofold. On one hand, the larger complexity of the morphology may result in a larger set of interactions between the VSR and the environment, that is, in a larger expressiveness. On the other hand, as already observed above, the larger search space of this case may take longer to converge to a good controller; i.e., from another point of view, within the same number of generations, different evolutionary runs may follow different paths in the search space which do not end in the same "point".

5.4.2 Environment: uneven surface

For the purpose of answering (RQ3), we considered a second case in which some aspect of the environment changes over the time. Differently than in the environment with even surface, variable environmental conditions constitute an opportunity for the sensing controller to exploit its peculiar ability of sensing the environment: that ability is instead not available for VSRs with the non-sensing controller.

For easing the experimentation, we introduced the variable conditions as a varying vector for the gravity acceleration. In particular, we varied the direction of the gravity vector during the simulation and kept constant its norm $\|\boldsymbol{g}\| = 9.8\,\mathrm{m\,s^{-2}}$. The condition can be expressed as a function describing the value of the x-component $g_x(k)$ of the gravity vector $\boldsymbol{g}$ over the time—assuming that the y-component is always equal to 0.

We proceeded as follows. First, we performed the evolutionary runs imposing a sinusoidal signal for the x-component of the gravity:

$$g_x^{\text{evo}} = \sin\left(2\pi f_{g^{\text{evo}}} k\Delta t\right) \tag{5.4}$$

TABLE 5.3: Fitness of the best individual at the end of the evolution and its traveled distance in the validation scenarios (both in mm, mean μ and standard deviation σ across the 30 runs) in the uneven environment. ρ is the ratio between the traveled distance in the validation scenario and the fitness value.

		Non-sensing			Sensing		
	Morph.	μ	σ	ρ	μ	σ	ρ
Fitness	Worm	120	8		3050	358	
	Biped	78	19		873	180	
	Tripod	556	60		620	378	
Flat	Worm	138	5	1.15	3132	498	1.02
	Biped	73	30	0.93	2228	187	2.55
	Tripod	528	43	0.95	727	100	1.17
Step	Worm	111	56	0.92	3543	412	1.16
	Biped	69	20	0.88	1010	226	1.15
	Tripod	539	160	0.96	870	213	1.40
Sin	Worm	104	7	0.86	3194	281	1.04
	Biped	77	17	0.99	446	151	0.51
	Tripod	505	48	0.91	512	175	0.82

where $f_{g^{\text{evo}}=2\frac{1}{\Delta t n_{\text{sim}}}=1.43\,\text{Hz}}$. Then, we assessed each evolved controller (i.e., the best individual at the last evolution) in three different *validation scenarios*:

$$g_x^{\text{flat}}(k) = 0 \tag{5.5}$$

$$g_x^{\text{step}}(k) = \begin{cases} 0 & \text{if } k \leq \frac{n_{\text{sim}}}{2} \\ 3 & \text{otherwise} \end{cases} \tag{5.6}$$

$$g_x^{\text{sin}}(k) = \sin\left(5\pi f_{g^{\text{evo}}} k\Delta t\right) \tag{5.7}$$

By considering validation scenarios which are different from the one using during the evolution, we hence also assessed the generalization ability of the EA in evolving the VSR controllers. Note that varying the direction of the gravity vector basically corresponds to considering an uneven, instead of flat, surface on which the VSR moves.

Table 5.3 shows the results of the experiments in the uneven environment.

It can be seen that, also in this environment, the sensing controller is always more effective than the non-sensing one. VSRs moved by the former travel a longer distance in any condition: both when computing the fitness (i.e., with g_x^{evo}) and in the validation scenarios (i.e., with g_x^{flat}, g_x^{step}, and g_x^{sin}). As for the even environment, differences are in general less apparent for the tripod than for the other two morphologies. All the differences are statistically significant according to the Mann-Whitney U-test ($\alpha = 0.05$): we do not show the values in the table.

Of more interest are the findings concerning the comparison between the fitness of the best individual and its performance in the validation scenario. Table 5.3 captures the outcome of this comparison in the two ρ columns: for a given morphology, controller, and validation scenario, ρ is the ratio of the distance traveled in the validation scenario and the fitness value, i.e., the one traveled with g_x^{evo}.

The key finding is that ρ is lower than 1 in most cases (8 on 9) for the non-sensing controller and greater than 1 in most cases for the sensing controller (7 on 9). VSRs

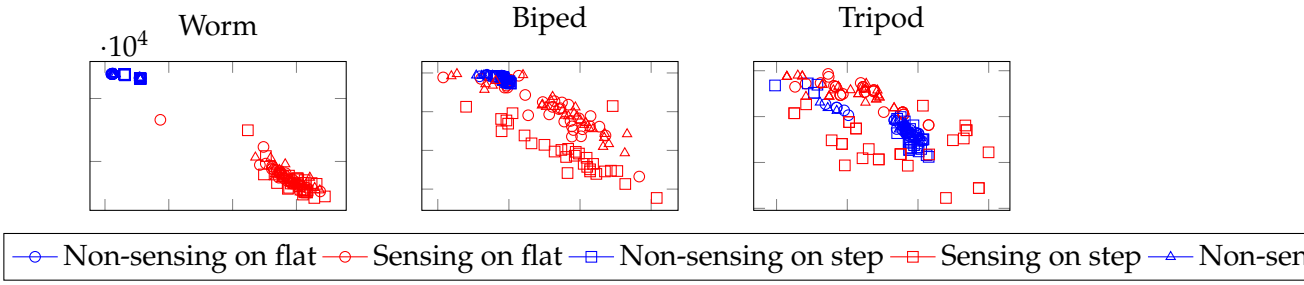


FIGURE 5.6: Behaviors resulting from the 60 best controllers evolved in the environment with uneven surface, with the three morphologies when testing them in the three validation scenarios.

equipped with the sensing controllers are hence able to move well on scenarios different than the one used for their evolution, whereas VSRs with non-sensing controller are not. We explain this clear difference with the fact that the sensing ability allows to react to an environment different from the one the controlled evolved and to adapt the VSR behavior.

Finally, Table 5.3 shows that, not surprisingly, the Sin validation scenario is the most difficult for all the VSRs: still, the worm equipped with a sensing controller is able to perform not worse on this scenario than on the one seen during the evolution ($\rho = 1.04$).

Analysis of the behaviors

We performed the same analysis of the behaviors as for the environment with the even surface. The results are shown in Figure 5.6.

The findings are similar to the previous case. Sensing controllers exhibit, in general, more various behaviors and this difference is less apparent for the tripod than for the worm and the biped. However, Figure 5.6 also highlights that the behaviors resulting from sensing controllers differ among the three validation scenarios. The difference is more apparent for the biped. We motivate this latter finding with the fact that this morphology is a good trade-off in complexity: it is not too simple to prevent large variation in the behaviors (like the worm), nor too complex to make harder the evolution of controller able to exhibit a well-defined behavior (like the tripod).

5.5 Conclusions

Voxel-based soft robots are a promising framework in which the behavior of a robot is determined by both its brain, i.e., its controller, and its body. In this work we have explored a form of holistic design in which the controller is equipped with sensing capabilities distributed across the full body of the robot. We have considered a sensing controller represented as a neural network and have considered the problem of synthesizing such a controller automatically, by means of an Evolutionary Algorithm. We have exercised such an algorithm on three different bodies, each in two different environments, with the aim of solving a locomotion task. We have compared the resulting sensing controller to a more traditional one, also synthesized automatically with the same Evolutionary Algorithm, and we have found that the sensing controller is more effective than its non-sensing counterpart, also when immersed in an environment different from the one in which it evolved.

We believe these results are very promising and suggest that the shifting of complexity from the controller to the body intrinsic to voxel-based soft robots, should be carefully coupled with forms of distributed sensing. We intend to investigate the potential of sensing controllers on larger robots and more complex tasks. In order to cope with the resulting complexity of the search space, we plan to rely on a more efficient evolutionary framework, such as, e.g., CPPN-NEAT, as well as a modular design in which robots are assembled out of smaller (parts of) robots evolved separately.

Chapter 6

Voxel-Based Soft-Robots Optimization towards Criticality

The development of voxel-based soft robots has allowed to shift the complexity from the control algorithm to the robot body itself, which is extremely versatile and more adaptable than the one of traditional robots, since it is made of soft materials. Nonetheless it is still not clear which are the factors responsible for the adaptability of the body, which we define as the ability to cope with tasks requiring different skills. In this work we propose a task-agnostic approach for automatically designing adaptable soft robotic bodies, based on the concept of self-organized criticality. This is a property belonging to systems close to a phase transition between the ordered and the chaotic regime. We let these bodies evolve towards self-organized criticality, and we then validate the impact of this approach on the actual adaptability by measuring the resulting bodies performance on three different tasks designed to require different skills. The validation results confirm that self-organized criticality is indeed one of the factors responsible for the adaptability of a soft robotic body, and that this is a promising approach for adaptability assessment.

6.1 Introduction

Traditionally, engineers have designed robotic systems modeled by connected joints made of rigid materials. These rigid-body robots can be programmed to efficiently perform a single task in a predictable way, but often with limited adaptability (Rus and Tolley, 2015).

Soft robots, on the contrary, are designed using soft material, in order to mimic nature in the way they interact with the environment. Since they are made of soft materials, soft robots are provided with almost infinite degrees of freedom and thus are capable of more natural movements. This allows a variety of different behaviors that were not possible with traditional robots, such as greater adaptability and many new opportunities for robotics (Lipson, 2014). Soft robotic bodies are able to bend and twist with high curvatures, through deforming part of their body in a continuous way (Mazzolai et al., 2012), and thus can move in confined spaces. Since they can adapt their body shape to the environment, soft robots are thus able to manipulate objects, move on rough terrain, and finally execute rapid and agile manoeuvres underwater. However, due to this complexity within their body, the design and control of soft robot is a challenging task, which suggests that traditional robotics techniques might not be effective.

Among the existing categories, Voxel-based Soft Robots (VSR) are made of many elastic blocks called voxels, defined by mechanical properties similar to those of biological tissues, which allow them to expand or contract when controlled by an

external signal. A VSR is defined by a body, which is simply an aggregate of voxels, and a brain, which is the control algorithm responsible for actuating its body. Biological inspired meta-heuristics such as Evolutionary Computation (EC) have been extensively applied (Hiller and Lipson, 2011; Cheney et al., 2013; Cheney, Bongard, and Lipson, 2015) and it has showed to be a promising approach for the design and control of VSR. However the optimization of these robots is often oriented towards a specific task, i.e. a locomotion task, which provides no guarantee on the effectiveness of the resulting bodies when subjected to a different task.

In this work we explore the possibility of automatically designing adaptable soft-robotic bodies by means of EC, such that the resulting bodies are able to successfully accomplish tasks requiring different motor skills. A simple way to find such bodies might proceed by evaluating each candidate solution on all the given tasks, and optimizing the body towards the maximization of the overall performance. This approach does not work in general, because we need to know in advance all the possible tasks, and the definition of a new one would make the results of the optimization pointless.

We propose here a task-agnostic approach for automatically designing adaptable bodies without requiring any information on the tasks, and instead based on the definition of self-organized criticality (Bak, Tang, and Wiesenfeld, 1988). In this approach we provide a criticality score value, based on the fitting of the empirical avalanches distribution (Bak, Tang, and Wiesenfeld, 1988), coming from the assessment of a body, with a target distribution. Given this fitness score, we then let the EC optimize the body towards higher values of this score.

To validate the bodies resulting from the evolution, we design three reasonably different tasks: a locomotion task on a plain ground, a jump task, and a task of escape from a narrow cave-like environment, and we test the bodies on these tasks by optimizing their controller on each task. Several other bodies have been considered for this validation, some of them inspired by previous works such as Talamini et al., 2019, while others automatically generated thorough algorithms based on randomness. We compare all the robots on these tasks, and we draw an overall ranking that show that self-organized criticality is a relevant aspect for the design of robots that are among the most adaptable. We notice also that designing bodies by means of an random algorithm, in some cases allows to obtain bodies with comparably high adaptability. Finally we notice that bodies designed to perform optimally on a single task exhibit almost no adaptability.

6.2 Related work

Seeking radical breakthroughs towards more adaptable machines and inspired by the biological systems, engineers have developed robots made of soft materials (Trimmer, 2013), which interact with the environment in a more natural way.

According to Rus and Tolley, 2015, these robotic systems are more adaptable and versatile than traditional ones made of rigid joints. However, this aspect has never been completely explored, since most of the research in the literature has dealt with the automatic design of robots for a very specific tasks: in Kriegman, Cheney, and Bongard, 2018 and Cheney et al., 2013 different aspects of soft robots such as the lifetime development and an effective representation are consider and the results are evaluated on their locomotion performance, in Sadeghi, Mondini, and Mazzolai, 2017 a soft robot inspired by the plants is engineered for growing, and finally in Shen,

Na, and Wang, 2017 a cephalopod molluscs inspired robot is developed to exhibit interesting underwater propulsion and manouvering mechanisms.

Reservoir Computing (RC) is a computational framework derived from echo state networks (Jaeger, 2002) and liquid state machines (Fernando and Sojakka, 2003), which leverages on the high-dimensionality of a dynamical system, i.e. its substrate, to produce training-efficient machine learning algorithms. Another remarkable property of RC is that the requirements for computationally powerful reservoirs turn out to be rather general and belonging to many different systems, as demonstrated in Schrauwen, Verstraeten, and Van Campenhout, 2007. Recent works on Reservoir Computing (RC) such as Li et al., 2012; Tanaka et al., 2019 and Nakajima et al., 2013, suggest that soft robotic systems, thanks to the intrinsically high-dimensionality, non-linearity, and elasticity proper of soft materials, which lead to overall highly complex and time-varying dynamics under actuation, are perfect substrates for RC. Specifically these works (Li et al., 2012; Nakajima et al., 2013) show that the structure of an octopus arm inspired soft robot can be exploited as a computational resource.

Self-organized criticality (SOC) is a property typically observed in slowly driven non-equilibrium systems with many degrees of freedom and strongly nonlinear dynamics (Bak, Tang, and Wiesenfeld, 1988), which naturally evolve towards a critical point of phase transition between chaotic and non-chaotic regimes (Bertschinger and Natschläger, 2004). Being close to this phase transition allows systems in which complex computation is possible, as presented in Langton, 1990, where the authors investigate the conditions that allow computation in Cellular Automata (CA). SOC is also inherently connected to the separation property, used in Gibbons, 2010 for the assessment of neural reservoirs, and it has been demonstrated by Brodeur and Rouat, 2012 and successfully by Heiney et al., 2019 for spiking neural networks, that the regulation of a RC substrate towards SOC allows to obtain a more powerful reservoir.

In this work we suggest that self-organized criticality is responsible for guiding the design of a soft robotic systems capable of complex and versatile behaviors, and we provide a metric for self-organized criticality that estimates how close is the empirical avalanches distribution (Bak, Tang, and Wiesenfeld, 1988) observed in a robot body to a target power-law distribution, where we use a convenient representation for the distributions based on the method explained in Clauset, Shalizi, and Newman, 2009. We consider the simulation tool for the optimization of 2-D voxel-based soft robots described and validated in Medvet et al., 2020b; Medvet et al., 2020a, and we let the EC find the body that maximize the SOC score, where the estimate of the avalanches spatial extension is based on the approach presented in Heiney et al., 2019 for biological neural networks as RC substrate, and the fitness function is inspired by the empirical estimate for evolving CA towards SOC done in Pontes-Filho et al., 2020. We consider also other bodies, some of from Talamini et al., 2019, and some others based on randomness, and we compare the adaptability of the bodies on three different tasks.

6.3 Model

6.3.1 Voxel-based Soft Robot

We assume a discrete time physics model in which time scale through regular intervals of time δt. Within this physics model, a *voxel* at time t is defined as $v^{(t)} = \left(x, y, a^{(t)}, \boldsymbol{i}^{(t)}\right) \in V$, where V is the space of all possible voxels states, $(x, y) \in \{0, n_{\text{grid}}\}^2$ are the 2-D coordinates of the voxel withing a grid of side n_{grid}, $a^{(t)} \in$

$[a_{\min}, a_{\max}]$ indicates the current ratio of the voxel area w.r.t. its resting one, and $\boldsymbol{i}^{(t)} \in \mathbb{R}^{n_{\text{sens}}}$ is the list of inputs collected at time t from the n_{sens} sensors.

We define a *robot* made of n_{voxel} voxels at time t as the pair $\boldsymbol{p}^{(t)} = \left(\boldsymbol{p}_{\text{body}}^{(t)}, \boldsymbol{p}_{\text{brain}}\right)$, where $\boldsymbol{p}_{\text{body}}^{(t)} = \left(v_1^{(t)}, \ldots, v_{n_{\text{voxel}}}^{(t)}\right) \subseteq V^{n_{\text{voxel}}}$ is a set of connected voxels at time t, and $\boldsymbol{p}_{\text{brain}} : V^{n_{\text{voxel}}} \times \mathbb{R}^{n_{\text{param}}} \times \mathbb{R} \mapsto V^{n_{\text{voxel}}}$ is the brain of the robot, defined by n_{param} real-valued parameters list $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_{n_{\text{param}}})$, such that $p_{\text{body}}^{(t+1)} = p_{\text{brain}}^{(t)}(p_{\text{body}}^{(t)}, \boldsymbol{\theta}, t)$ provides a control signals for each voxel at a given time t, and actuates the body accordingly.

A body of $_{\text{voxel}}$ voxels is connected if the following property holds $\forall u \in \{1, \ldots, n_{\text{voxel}}\}$, such that:

$$\exists v \in \{1, \ldots, n_{\text{voxel}}\}, v \neq u : (|x_u - x_v| = 1 \wedge y_u = y_v) \vee (|y_u - y_v| = 1 \wedge x_u = x_v) \tag{6.1}$$

We formalize the body initializer as a function $\phi : \mathbb{R}^2 \mapsto V^{n_{\text{voxel}}}$, which given the pair $\left(n_{\text{grid}}, n_{\text{voxel}}\right)$ at time $t = 0$ initializes a body $\boldsymbol{p}_{\text{body}}$, where we omit the time reference for the sake of simplicity. For each voxel of $\boldsymbol{p}_{\text{body}}$, it holds that its area ratio at time $t = 0$ is initialized as *resting*, that is $a^{(0)} = 1$, and its 2D coordinates (x, y) are statically and uniquely defined.

Simulating the robot $\boldsymbol{p}^{(t)}$ at each time t means applying the function $\boldsymbol{p}_{\text{brain}}$ to the body $\boldsymbol{p}_{\text{body}}^{(t)}$, which updates its state at the next time $t + 1$ into $\boldsymbol{p}_{\text{body}}^{(t+1)}$.

6.3.2 Adaptability

A voxel of the robot $\boldsymbol{p}^{(t)}$ is called *active* at time t if it satifsy the condition on the area ratio $|a^{(t)} - a^{(t-1)}| > \tau_{\text{active}}$, where τ_{active} it a dynamically computed threshold based on n_{voxel} of robot $\boldsymbol{p}^{(t)}$.

We define the spatial extension $s \in \{0, n_{\text{voxel}}\}$ of an *avalanche* of a robot which body is initialized by $\phi(n_{\text{voxel}}, n_{\text{grid}}) = \boldsymbol{p}_{\text{body}}^{(0)}$, and is simulated for T_{adapt} time, as the count of the voxels of $\boldsymbol{p}_{\text{body}}^{(0)}$ active at least once from $t = 0$ to $t = T_{\text{adapt}}$, in which $\boldsymbol{p}_{\text{brain}}$ actuates one voxel of the body with a pulse control signal at time $t = 0$.

By initializing and simulating from $t = 0$ to $t = T_{\text{adapt}}$ the same body n_{voxel} times, each time actuating a different voxel with a pulse control signal at time $t = 0$, it is possible to collect an *empirical* avalanches distribution $\mathcal{S}$, which is an approximation of a theoretical one. Through the notion of self-organized criticality, we define the degree of *adaptability* of a body, as the proximity of the corresponding avalanches distribution $\mathcal{S}$ to a power-law αx^{-k}, defined by parameters α, k, and independent variable x. For an adaptable body, it holds that $\mathcal{S} \sim \alpha x^{-k}$, that is $\mathcal{S}$ is smpled from a power-law distribution.

6.4 Experiments

6.4.1 Adaptability

Activity threshold

The design of the activity threshold τ_{active} is extremely important for the correct estimate of adaptability of a body, since some τ_{active} do not even allow to satisfy the adaptability condition.

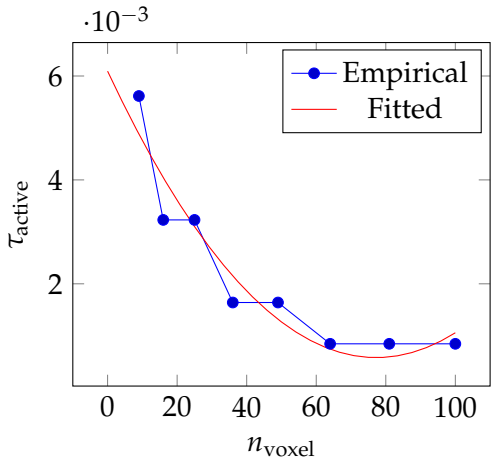


FIGURE 6.1: Polynomial fitting of the activity threshold τ_{active} corresponding to the highest variance avalanches distribution for each value of n_{voxel}.

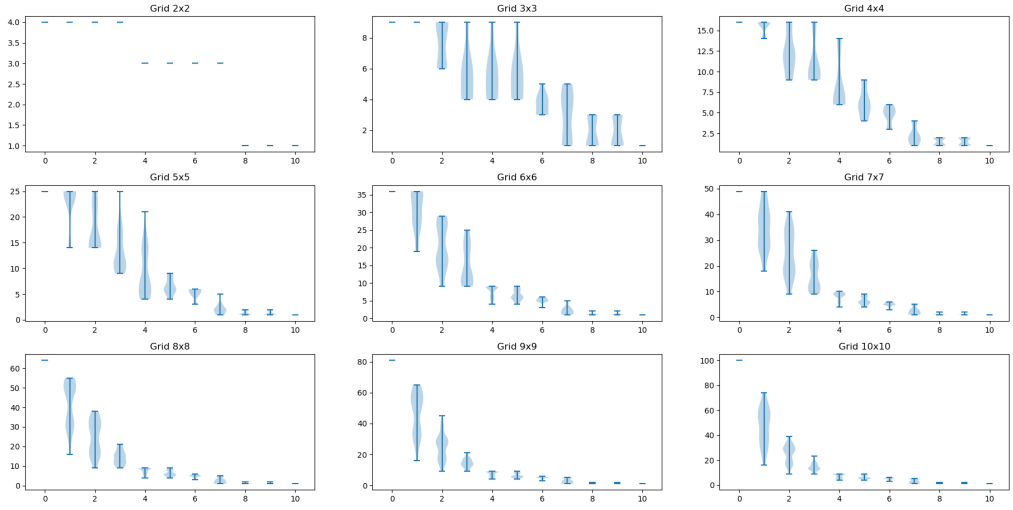


FIGURE 6.2: Each plot shows n_{thr} avalanches distributions obtained with a different activity threshold $\tau_{\text{active}} \in [\tau^{\text{all}}_{\text{active}}, \tau^{\text{one}}_{\text{active}}]$, given a n_{grid} value.

We call $\tau^{\text{one}}_{\text{active}}$ the value of activity threshold that produces always avalanches of spatial extension 1, which means that only the voxels that are actuated with the pulse are affected by the avalanches. On the other hand, we call $\tau^{\text{all}}_{\text{active}}$ the value of activity threshold that produces always avalanches of spatial extension n_{voxel}, which means that all the voxels are affected by the avalanches.

Ideally the optimal value of τ_{active} is the one that produces the avalanches distribution with the greatest variance for all the possible n_{voxel} values.

To meet this condition, we consider all the robots with square bodies initialized with $n_{\text{grid}} \in \{n^{\text{min}}_{\text{grid}}, n^{\text{max}}_{\text{grid}}\}$, and where each body is made of $n_{\text{voxel}} = n^2_{\text{grid}}$, and we collect n_{thr} avalanches distributions, each one corresponding to a different value of $\tau_{\text{active}} \in [\tau^{\text{all}}_{\text{active}}, \tau^{\text{one}}_{\text{active}}]$. Each plot in Figure 6.2 show the resulting distributions for a specific value of n_{grid}, where each distribution correspond to a different value of τ_{active}. As n_{grid} increases, the avalanches distributions with highest variance within the plots in Figure 6.2 shift towards lower threshold values.

Among the n_{thr} values of activity threshold, we pick one for each robot, corresponding to the avalanches distibutions with the greatest variance, which results in the list of pairs $\left(((n^{\text{min}}_{\text{grid}})^2, \tau^1_{\text{active}}), \ldots, ((n^{\text{max}}_{\text{grid}})^2, \tau^{n_{\text{thr}}}_{\text{active}})\right)$, which we represent as blue dots in Figure 6.1. Given these dots, we perform a degree-two polynomial interpolation, which provide us an empirical way to estimate the optimal activity threshold for a givben body of n_{voxel} voxels, as $\tau^{n_{\text{voxel}}}_{\text{active}} = \lambda_1 + \lambda_2 n_{\text{voxel}} + \lambda_3 n^2_{\text{voxel}}$.

Adaptability assessment

When we place the robots on the ground for assessing their adaptability, the impact of external forces, i.e. gravity, might affect the simulation by introducing undesired dynamics which have nothing to do with the avalanches estimate. In order to circumvent this problem, while avoiding to waste computational time waiting for this dynamic to end, we simulate the robots in absence of gravity.

To assess the adaptability of a body $p_{\text{body}} = \phi(n_{\text{grid}}, n_{\text{voxel}})$, we simulate the robot for n_{voxel} times, each time for T_{adapt} time steps, with activity threshold $\tau^{n_{\text{voxel}}}_{\text{active}}$, and we collect the corresponding empirical avalanches distribution $\mathcal{S}$. An empirical distribution fits a power-law distribution if the log-log plot of this distribution is a line with negative slope. Therefore we compute the log-log of $\mathcal{S}$, which we call $\mathcal{S}'$, and we fit it with a linear regression, that we call the underlying *theoretical* distribution $\mathcal{Q}$.

We consider $f_{\text{det}} : \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|} \mapsto [0,1]$, which computes the coefficient of determination (Nagelkerke et al., 1991) from the distributions $\mathcal{S}'$ and $\mathcal{Q}$, and the $f_{\text{ks}} : \mathbb{R}^{|\mathcal{S}|\times|\mathcal{S}|} \mapsto \mathbb{R}$, which computes the Kolmogorov-Smirnov statistic (Drew, Glen, and Leemis, 2000), and we conclude that the degree of adaptability of the body that has generated $\mathcal{S}$ is $f_{\text{adapt}} = (e^{-f_{\text{ks}}})^2 + f_{\text{det}} \in [0,2]$.

Adaptability optimization

We consider a population-based evolutionary algorithm, where each individual *phenotype* is a robot body $\boldsymbol{p}_{\text{body}} = \phi(n_{\text{grid}}, n_{\text{voxel}})$, internally represented by its *genotype* $\boldsymbol{g} \in [0,1]^{n^2_{\text{grid}}}$.

The genotype-phenotype mapping function $m : [0,1]^{n^2_{\text{grid}}} \mapsto V^k$, for each i-th element of the genotype, if $g_i > \tau_{\text{map}}$, places a voxel $\boldsymbol{p}^{(0)}_{\text{body}} \cup \{v^{(0)}(\frac{i}{n_{\text{grid}}}, i \mod n_{\text{grid}}, a^{(0)}, i^{(0)})\}$ in the body, where τ_{map} is the mapping threshold
$\tau_{\text{map}} = \arg\max_j \left(\left|\bigcup_{i=1}^{|\boldsymbol{g}|}\{g_i > g_j\}\right| \geq n_{\text{voxel}}\right)$.

The fitness function is the value f_{adapt} of adaptability of the body.

A population of n_{pop} individuals is randomly sampled, and evolved by means of genetic operators, namely uniform crossover and gaussian mutation with probability p_{mut}, a tournament selection is applied with size n_{tourn}, and the evolutionary loop is repeated for n_{iter} iterations. We perform n_{run} experimental runs and we collect the best performing body from each of them.

6.4.2 Validation

Controllers

We consider 2 controllers variants:

- Phase controller $\boldsymbol{p}_{\text{brain}}^{\text{phase}}$, is a stateless non-sensing brain, in which the control signal is function of t and of the $n_{\text{param}} = n_{\text{voxel}}$ parameters, each corresponding to the phase of the sine wave applied to one voxel.
- Neural network controller $\boldsymbol{p}_{\text{brain}}^{\text{nn}}$, is a stateful sensing brain in the form of a neural network with $n_{\text{voxel}} n_{\text{sens}}$ inputs, one hidden layer of $0.65 n_{\text{voxel}} n_{\text{sens}}$ neurons and hyperbolic tangent activation function, and n_{voxel} outputs. The control signal is function of t, of the $n_{\text{param}} = 0.65 n_{\text{voxel}}^2 n_{\text{sens}}(1 + n_{\text{sens}})$ parameters of the neural network, and of the inputs $\left(\boldsymbol{i}_1^{(t)}, \dots, \boldsymbol{i}_{n_{\text{voxel}}}^{(t)}\right)$ coming from each voxel sensors at time t.

Tasks

We consider 3 *tasks*, where each one is defined within a different environment, by its own length expressed in time steps, and by task-specific metrics used for the assessment:

- Locomotion on a flat ground, in which each brain is evaluated upon the fitness $f_{\text{loc}} : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$, difference between the x-coordinate of the center of mass of its body at time T_{loc} from the one at time $t = 0$.
- Jump, in which each brain is evaluated upon the fitness $f_{\text{jmp}} : \mathbb{R}^{\frac{T_{\text{jmp}}}{\delta t}} \mapsto \mathbb{R}$ maximum height reached w.r.t. the initial position of its body, considering the trajectory along the y-axis of the center of mass from $t = 0$ to T_{jmp}, with a time step δt.
- Escape from narrow environment, in which each brain is evaluated upon the fitness $f_{\text{esc}} : \mathbb{R} \times \mathbb{R} \mapsto \mathbb{R}$, difference between the x-coordinate of the center of mass of its body at time T_{esc} from the one at time $t = 0$.

Baselines

We consider a grid of size n_{grid}, and a desired number of voxels n_{voxel}, and we define the following body initializer baselines:

- Pseudo-random initializer $\phi_{\text{ps-rnd}}$, which starts from generating an initial randomly positioned voxel, and recursively adds a new voxel to the body in a random position next to the most recently added one, until the number of voxels in the body is n_{voxel}. This initializer often generates solid bodies (second row in Figure 6.3), where all the voxels are close to each others.

TABLE 6.1: Experiments parameters.

	Description	Parameter	Value
Sim.	Minimum area ratio	$a_{\min}$	0.75
	Maximum area ratio	$a_{\max}$	1.25
	Time step length	δt	0.1
Adapt.	Grid side	n_{grid}	20
	Voxels in a body	n_{voxel}	20
	Activity thr. all voxels	$\tau_{\text{active}}^{\text{all}}$	0.008
	Activity thr. one voxels	$\tau_{\text{active}}^{\text{one}}$	0.00005
	Threshold sampled	n_{thr}	11
	Avalanche estimate time	T_{adapt}	30
	Population size	n_{pop}	1000
	Iterations	n_{iter}	200
	Mutation probability	p_{mut}	0.01
	Tournament size	n_{tourn}	10
	Experimental runs	n_{run}	10
	alpha1		
	alpha2		
	alpha3		
Valid.	Locomotion assessment time	T_{locom}	20
	Jump assessment time	T_{jump}	20
	Escape assessment time	T_{escape}	40
	Births	n_{birth}	10000

- Random initializer ϕ_{rnd}, which iteratively creates new voxels with uniformly sampled coordinates, until the number of voxels in the body is at least n_{voxel}. This initializer often generates irregular bodies of unpredictable sizes (third row in Figure 6.3), and these bodies exhibit on average higher adaptability w.r.t. the ones generated with ϕ_{psrnd}.

We consider also bodies manually initialized using the domain knowledge (fourth row in Figure 6.3). The *worm* and the *biped* robots have been inspired by the results of Talamini et al., 2019; Medvet et al., 2020a; Medvet et al., 2020b, while the *box* and *revT* are some of the simplest body that can be manually designed. All these robots are made of exactly n_{voxel}, and they all exhibit a low adaptability score.

Controller optimization

We consider an evolutionary algorithm based on a Covariance Matrix Adaptation Evolutionary Strategies (CMA-ES) (Hansen, Müller, and Koumoutsakos, 2003), where each individual phenotype represents the n_{param} parameters of a robot brain, and is internally represented by its genotype $\boldsymbol{g} \in [0, 1]^{n}_{\text{param}}$.

The genotype-phenotype mapping is the identity function, and the fitness function is one among $(f_{\text{loc}}, f_{\text{jmp}}, f_{\text{esc}})$, depending on the specific task.

A number of individuals n_{birth} are sampled from a multivariate gaussian distribution using CMA-ES, and evolved by means of genetic operators. We perform n_{run} experimental runs, and we collect the best performing brain from each of them.

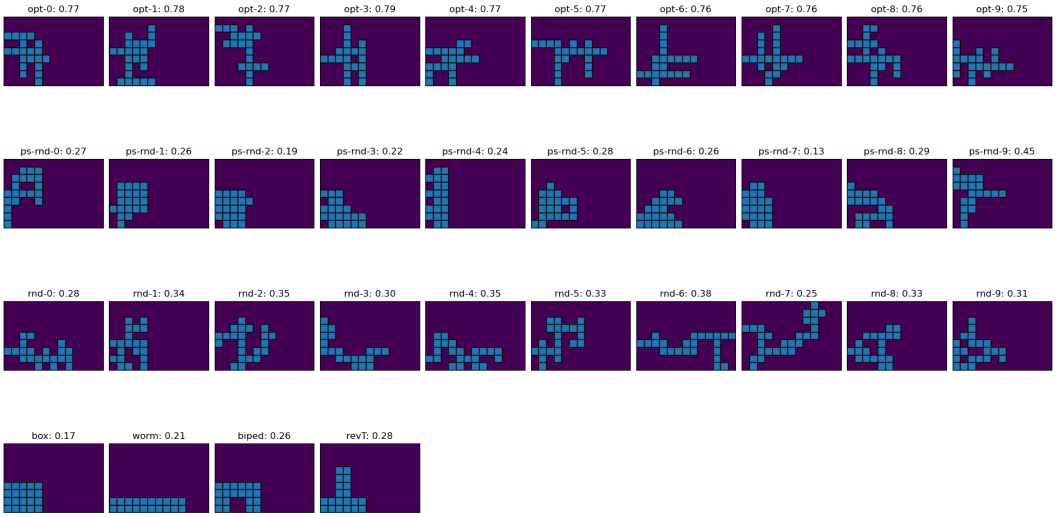


FIGURE 6.3: Value of criticality for all the different bodies.

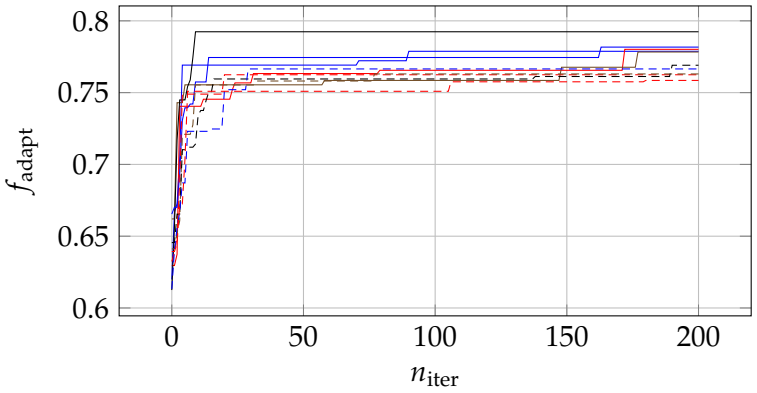


FIGURE 6.4: Adaptability of the best performing individual of the current population during n_{iter} evolutionary iterations among n_{run} experimental trials.

6.5 Results

6.5.1 Adaptability

The configuration employed in this experimental campaign can be seen in Table 6.1. The results of the n_{run} experimental runs for evolving the robot bodies towards the adaptability condition can be seen in Figure 6.4. From these results it can be seen that, despite the use of a dynamic activity threshold, there is still an intrinsic boundary which prevents the evolution to find a body with maximum adaptability value. However these bodies obtained from the evolutionary process, (first row in Figure 6.3) have reached higher adaptability values than all the other baselines. By looking at Figure 6.3, it seems that, despite being radically different, all these bodies have all developed some limbs resembling arms or legs. This aspect is missing in all the baselines, and is probably relevant for the higher adaptability score.

TABLE 6.2: Validation results comparing the bodies optimized for adaptability and the manually designed ones using a phase controller $\boldsymbol{p}_{\text{brain}}^{\text{phase}}$. All the values in the table are multiplied by 10^3.

	Locomotion			Jump			Escape			Overall	
Rank	Body	f^*_{loc}	σ	Body	f^*_{jmp}	σ	Body	f^*_{esc}	σ	Body	f_{adapt}
1	worm	6.31	0.6	**opt-1**	3.03	0.3	opt-2	1.64	0.4	**opt-1**	0.78
2	biped	5.96	1.3	biped	2.85	0.3	opt-6	1.20	3.2	biped	0.26
3	revT	4.44	5.2	box	2.50	0.4	**opt-1**	0.82	0.3	opt-9	0.75
4	box	4.41	1.0	worm	2.40	0.6	opt-9	0.79	0.1	worm	0.21
5	**opt-1**	2.54	0.2	revT	2.23	0.2	opt-8	0.75	0.1	revT	0.28
6	opt-4	2.41	0.6	opt-9	2.14	0.4	opt-0	0.73	0.1	opt-6	0.76
7	opt-9	2.33	0.6	opt-3	2.14	0.3	opt-7	0.67	0.1	box	0.17
8	opt-5	2.27	0.5	opt-6	2.03	0.4	opt-5	0.52	0.2	opt-2	0.77
9	opt-6	2.16	0.5	opt-5	1.69	0.3	opt-3	0.43	0.2	opt-5	0.78
10	opt-2	1.81	0.5	opt-0	1.58	0.2	opt-4	0.42	0.2	opt-4	0.77
11	opt-3	1.20	0.2	opt-4	1.57	0.3	revT	0.34	0.0	opt-3	0.79
12	opt-7	1.01	0.4	opt-7	1.46	0.1	biped	0.08	0.0	opt-0	0.78
13	opt-0	0.93	0.4	opt-2	0.79	0.0	box	0.04	0.0	opt-7	0.77
14	opt-8	0.72	0.1	opt-8	0.74	0.4	worm	0.04	0.0	opt-8	0.77

6.5.2 Validation

Manually designed baselines

In Table 6.2 we present the result of the validation of the bodies optimized towards adaptability, where each *i*-th body is indicated as *opt-i*, and the manually engineered ones, which we call *worm*, *biped*, *revT*, and *box*. These results are obtained by optimizing n_{run} possibly different phase controllers for each body considered, and for each task. Specifically, the values of f^*_{loc}, f^*_{jmp}, and f^*_{esc} showed in this table represent the median value of the best fitness value among the n_{run} experimental trials, for each of the 3 tasks.

Considering only the results on the locomotion task, it is clear from the table 6.2 that the manually designed bodies, namely *worm*, *biped*,*revT*, and *box*, are actually the 4 most effective ones, in terms of the value of f^*_{loc} they achieve. However, it can be also seen from Table 6.2 that some of the manually designed bodies, while being often extremely effective on a specific task, i.e. the *worm* robot is the best performing one on the locomotion task, they might be slightly less performing or even the less effective ones on other tasks, i.e. the *worm* robot is the least effective one considering the value of f^*_{esc} measured on the escape task. Therefore we claim that the manually designed robots are therefore hardly adaptable, and this is confirmed also by their extremely low f_{adapt} score.

To better understand this aspect, we compute the average ranking occupied by each body, and we show the overall ranking on the rightmost part of Table 6.2, and we use this value as a proxy for the validation of the adaptability. This ranking suggests that the overall most adaptable body is therefore *opt-1*, thus confirming that the approach proposed allows to generate adaptable robotic bodies.

TABLE 6.3: Validation results comparing the bodies optimized for adaptability and the randomly designed ones using a phase controller $p_{\text{brain}}^{\text{phase}}$. All the values in the table are multiplied by 10^3.

	Locomotion			Jump			Escape			Overall	
Rank	Body	f^*_{loc}	σ	Body	f^*_{jmp}	σ	Body	f_{es}	σ	Body	f^*_{adapt}
1	rnd-4	3.02	0.9	rnd-5	3.37	1.0	psrnd-6	2.09	0.1	rnd-4	0.35
2	rnd-0	2.97	0.3	**opt-1**	3.03	0.3	psrnd-5	1.66	0.2	rnd-5	0.34
3	rnd-9	2.94	0.3	rnd-4	2.89	0.8	opt-2	1.64	0.4	**opt-1**	0.78
4	rnd-5	2.86	0.5	opt-9	2.14	0.4	psrnd-8	1.60	0.2	opt-6	0.76
5	**opt-1**	2.54	0.2	opt-3	2.14	0.3	psrnd-3	1.58	0.3	rnd-9	0.31
6	rnd-8	2.46	0.4	rnd-1	2.06	0.3	rnd-3	1.23	0.4	psrnd-6	0.26
7	opt-4	2.41	0.6	opt-6	2.03	0.4	rnd-4	1.21	0.2	opt-9	0.75
8	opt-9	2.31	0.7	rnd-0	2.00	0.5	opt-6	1.20	0.3	psrnd-5	0.28
9	opt-5	2.27	0.5	rnd-9	1.98	0.3	psrnd-2	1.16	0.1	rnd-0	0.28
10	opt-6	2.16	0.5	rnd-2	1.71	0.5	rnd-7	1.15	0.1	psrnd-8	0.29
11	psrnd-6	2.03	0.2	opt-5	1.69	0.3	rnd-5	1.07	0.3	opt-2	0.77
12	rnd-1	1.90	0.3	psrnd-5	1.66	0.2	psrnd-0	1.07	0.1	rnd-1	0.34
13	opt-2	1.81	0.5	rnd-8	1.62	0.1	psrnd-4	1.03	0.1	psrnd-3	0.22
14	psrnd-5	1.54	0.2	opt-0	1.58	0.2	rnd-9	1.02	0.2	opt-5	0.78
15	psrnd-1	1.47	0.1	psrnd-6	1.58	0.2	**opt-1**	0.82	0.3	rnd-8	0.34
16	rnd-6	1.46	0.4	opt-4	1.57	0.3	opt-9	0.79	0.1	opt-4	0.77
17	psrnd-3	1.42	0.2	psrnd-0	1.46	0.2	psrnd-7	0.78	0.0	opt-3	0.79
18	psrnd-8	1.35	0.2	opt-7	1.46	0.1	psrnd-1	0.77	0.0	psrnd-0	27
19	opt-3	1.20	0.2	psrnd-8	1.42	0.2	opt-8	0.75	0.1	rnd-3	0.30
20	rnd-2	1.17	0.4	rnd-3	1.39	0.4	opt-0	0.73	0.1	psrnd-2	0.19
21	rnd-7	1.13	0.1	psrnd-3	1.32	0.2	psrnd-9	0.70	0.1	rnd-2	0.35
22	psrnd-0	1.01	0.1	psrnd-2	1.24	0.1	rnd-0	0.69	0.2	psrnd-1	0.26
23	opt-7	1.01	0.36	psrnd-7	1.18	0.19	opt-7	0.67	0.1	rnd-7	0.26
24	psrnd-2	1.01	0.0	psrnd-1	1.12	0.2	opt-5	0.52	0.2	opt-0	0.77
25	rnd-3	0.98	0.4	psrnd-4	0.89	0.0	rnd-1	0.51	0.1	opt-7	0.77
26	opt-0	0.93	0.4	opt-2	0.79	0.0	rnd-2	0.43	0.1	psrnd-4	0.25
27	psrnd-4	0.78	0.1	rnd-7	0.77	0.2	opt-3	0.43	0.2	psrnd-7	0.13
28	opt-8	0.72	0.1	rnd-6	0.77	0.2	opt-4	0.42	0.2	rnd-6	0.38
29	psrnd-7	0.65	0.0	opt-8	0.74	0.4	rnd-6	0.32	0.2	opt-8	0.77
30	psrnd-9	0.58	0.0	psrnd-9	0.54	0.0	rnd-8	0.20	0.0	psrnd-9	0.45

TABLE 6.4: Validation results comparing the bodies optimized for adaptability with the manually designed ones, and the ones generated with ϕ_{rnd} and ϕ_{psrnd}, using a neural sensing controller $\boldsymbol{p}^{\text{nn}}_{\text{brain}}$. All the values in the table are multiplied by 10^3.

	Opt-vs-manual		Opt-vs-random	
Rank	Body	f_{adapt}	Body	f_{adapt}
1	**opt-5**	0.78	**opt-5**	0.78
2	opt-7	0.77	rnd-5	0.34
3	opt-1	0.78	opt-1	0.78
4	opt-6	0.76	opt-7	0.77
5	opt-9	0.75	psrnd-0	0.28
6	opt-4	0.77	psrnd-9	0.46
7	worm	0.22	psrnd-1	0.26
8	opt-8	0.76	opt-6	0.76
9	revT	0.28	psrnd-5	0.28
10	biped	0.26	opt-9	0.76
11	opt-2	0.78	rnd-2	0.35
12	box	0.17	opt-4	0.77
13	opt-0	0.78	rnd-4	0.35
14	opt-3	0.79	psrnd-8	0.30

Adaptability approximation through randomness

Despite the promising results of the proposed strategy for optimizing body adaptability, the evolution towards self-organized criticality is a resources intensive process. Seeking for obtaining similar results with a less expensive approach, we consider two different body initialization algorithm based on randomness, namely ϕ_{rnd} and ϕ_{psrnd}, resulting respectively into the *i*-th body *rnd-i*, and *psrnd-i*. The results presented in Table 6.3 show that these two initialization strategies produce highly adaptable bodies. In particular, the ϕ_{rnd} generates bodies which in the case of *rnd-4* and *rnd-5* outperform even the ones evolved towards self-organized criticality, in terms of overall ranking. Looking at the Figure 6.3 we can see that these bodies on average show higher adaptability score w.r.t. the ones generated with ϕ_{psrnd}, which is indeed not as effective for generating adaptable bodies.

Neural sensing controller

We then further investigate the limitations of the proposed method based on the phase controller, by repeating the same validation process with a sensing controller $\boldsymbol{p}^{\text{nn}}_{\text{brain}}$ based on an artificial neural network. This control variant allows the robots to take even more advantage of the body complexity, by providing a number of sensors distributed through the body, and therefore it might privilege the most irregular bodies. In Table 6.4 we show the overall robots ranking, where in the first part of the table the values comes from considering the bodies resulting from optimization and the manually designed ones, and in the second part we compare the adaptable bodies and ones generated by means of ϕ_{rnd} and ϕ_{psrnd}. Looking at the results presented in Table 6.4 they support this claim, by showing that almost each *i*-th adaptable body *opt-i* is more adaptable than the manually designed ones. From the results showed in Table 6.4 it seems also that one of these bodies, namely *opt-5*, successfully elaborates the data collected by the sensors distributed through its body, allowing to reach the

highest ranking. This final result might suggest that the initializers ϕ_{rnd} and ϕ_{psrnd} are slightly less powerful when distributed sensing is employed.

6.6 Conclusions

In this work we have proposed an approach for measuring the adaptability of a soft robotic body towards different tasks, based on the definition of self-organized criticality. To this extent have proposed an algorithm for guiding the automatic design of adaptable soft robotic bodies by means of EC. We have then validated the resulting bodies on three tasks requiring different skills, against some manually designed bodies inspired by previous works, using a simple non-sensing control algorithm. The results show that this approach results into more adaptable robots, w.r.t. the manually crafted ones. Motivated by the computational cost of this optimization, we have considered other design algorithms based on randomness, which on the contrary are not computationally expensive, and allows to produce even more adaptable bodies. Finally we have repeated the validation process, now considering a sensing controller variant, which allows the bodies optimized for adaptability to leverage the body complexity. Provided with distributed sensing, these bodies reach the highest overall ranking w.r.t. any baseline.

Chapter 7

Assessing Actors in the Academic Publishing System

When it comes to analyze scientific production, some measurable quantities can be considered good proxies for understanding other non-measurable ones, whereas some other measurable quantities may not be a good choice. H-index and Impact factor are measurable quantities used to explain, respectively, the quality of an author and the quality of a journal. Are H-index and Impact factor good proxies? We investigate these research questions by designing a simulation of a simple scientific scenario that we let evolve over time, and we consider different strategies for authors and editors. In this simulated scenario we show that H-index seems to be a good predictor of an author quality, whereas the Impact factor as well as the number of papers published by a journal are not good predictors of the quality of that journal, except when a mild—but not too strict—scrutiny is applied by the editor.

7.1 Introduction

The results and the impact of a paper could be in practice be biased by a series of Questionable Research Practices (QRPs), i.e., a series of design choices, data, analysis, or results that should not be produced. These practices make the results of the papers in which they are employed impossible to be reproduced, and therefore invalidate their findings, and undermine their authors credibility. Regarding the problem of authors employing QRPs, the work by Agnoli et al., 2017 showed, through an extensive survey, that this kind of behavior could be generalized to different countries.

The peer review process is considered an important phase in the publication of an article, and ideally it is necessary to ensure the quality of a paper. Unfortunately the actors involved in this process may express a biased judgment, motivated by unethical reasons, like increasing the prestige of the program committees or the editorial boards in which they are involved or, in the case of publishers, the opportunity to make greater profit. Computer-generated published papers have been detected by Springer and IEEE in the past years (Van Noorden, 2014), thus proving that it is therefore possible to publish fake researches. In the age of AI, Bartoli et al., 2016 have investigated the feasibility of automatic generation of fake scientific reviews. This strategy allows the researchers to produce reviews without even reading the papers they should base the reviews on, and thus being involved in as many reviewing committees as possible, and it allows predatory publishers (Herron, 2017) to improve their credibility by sending many reviews to the authors.

7.2 Related work

Some publications have had a huge impact on the progress of sciences, and they have dramatically changed the direction of the future scientific researches. The value of these works has been recognized by most researchers and their authors have been awarded for their discoveries. Unfortunately these works are very few and distinct ones, while the majority of the other papers brings more limited innovations, and they represent a valuable contribution for only groups of researchers of a specific field Hirsch, 2005.

A fair and reliable metric for evaluating research papers impact and relevance for author belonging to the second category is therefore required (Hirsch, 2005). This quantification is necessary when more researchers have to be compared for university faculty recruitment and advancement, award of grants, etc. This quantification of the published papers might work as a proxy for evaluating also the authors who have produced those papers, and the journals who have published them as well. Specifically the authors are evaluated for the quality of the papers they have produced, and the journals are evaluated according to the quality of their published papers.

The necessity of finding the most suitable bibliometric indicators for scientific researches assessment has become even more relevant due to the growth of scientific literature in the last decades (Bornmann and Mutz, 2015).

7.2.1 Authors indicators

Many author-level bibliometric indicators have been proposed over time, as the demand of quantitative tools for assessing individual authors has increased during the last years.

A bibliometric index widely used for assessing the research output of individual authors is the well-known H-index, defined as the number of papers that have received as many citations as that number Hirsch, 2005. This index captures both the productivity of an author and the citation impact of his/her papers, irrespective of the publication venue of those papers and of the citing ones.

One obvious potential drawback of the H-index is that an highly productive author may strive to cite his/her own papers systematically: a moderate amount of papers with carefully chosen self-citations may boost the H-index of an author, irrespective of the actual quality of both the citing and cited papers. Indeed, the very same proposal by Hirsch, 2005 suggests to exclude self-citations from the computation of citations used for determining the H-index of an author.

Van Raan, 2006 has made a comparison between the H-index (and few other bibliometrics measures) and the peer reviewers judgment, received by the scientific output in the period 1991–2000. The individual results have been grouped w.r.t. the research group, i.e. university department they belong to, and the citations were limited to 3 a years window, instead of the entire lifetime. The results of this comparison have showed that H-index is indeed a reliable way to measure the peers judgment.

An overview of the existing bibliometric measures has been done by Todeschini and Baccini, 2016, where the authors have presented all the known bibliometric indicators in an encyclopedic form. Another significant work in this area is Wildgaard, Schneider, and Larsen, 2014, that has reviewed 108 indicators for evaluating the performances of scientific authors, by focusing on the ease of use of indicators as well as on the complexity of their calculations in relation to what they are supposed to reflect. An important contribution by the cited work is the distinction between

indicators that qualify the research output (on the author and on the publication venue) and indicators that measure the effects of output on the other researchers over time, that is the number of citations received.

Many criticisms have been made toward the practice of assessing authors mainly or solely by means of bibliometric indicators (Ruocco et al., 2017), including the fact that indicators may be used in a way that does not reflect their actual meaning, e.g., (Hicks et al., 2015; Corrall, Kennan, and Afzal, 2013; Barnes, 2017).

In this respect it has also been observed that the web and academic search engines have radically changed the behavior of researchers including, in particular, the choice of which papers to cite. Such choice is increasingly based on ease of access and discoverability by search engines, which may be loosely related to the actual quality of a paper, thereby making the practice of using the number of citations as a proxy for quality less justified (Bartoli and Medvet, 2014).

Another potential drawback of assessing authors by means of their bibliometric indicators, that is particularly relevant for our work, is that authors may modify their behavior in order to optimize those indicators at the expense of other features more tightly linked to the real quality of a research output, e.g., rigor, accuracy, and depth of the reviewing process of a publication venue. An important analysis in this respect has been provided by Baccini, De Nicolao, and Petrovich, 2019. The cited work defined a new *inwardness* indicator for a country, as the proportion of citations coming from authors in the same country, over the total number of citations gathered by the papers produced in that country. Through a comparative study on all the G10 countries in the period 2000–2016, this study has revealed that Italy was the country with the highest inwardness value. The key point is that this trend has started in the period 2011–2012, that is, exactly when new national regulations for personal careers of academic researchers introduced necessary conditions based mainly on the comparison between certain bibliometric indicators and a predefined threshold (Peroni et al., 2020).

7.2.2 Editors indicators

A key bibliometric index for scientific journals is the *impact factor*, defined as the ratio between the number of citations received, and the number of papers published in a 2 years window (Garfield, 1972). This quantitative index is widely used as a proxy for the scientific quality of the corresponding journal. It is often used also for quantitative assessment of research outputs by individual authors, on the assumption that all the papers published on a given journal are of the same the quality, as captured by the journal impact factor. Amin and Mabe, 2004; Andrés, 2009 analyzed the common usages of the impact factor and provided suggestions for preventing or mitigating the effects of its misuse.

In this respect, it is important to consider that the distribution of citations across papers in the same journal is highly skewed, and that this property holds regardless of the impact factor of that journal (Seglen, 1992), and therefore it follows that the impact factor of a journal is not representative of the quality of the individual articles.

Irrespective of the actual relation between impact factor of a journal and quality of the corresponding papers, the former has become a crucial tool for many academic communities (Ellegaard and Wallin, 2015) and it is important to remark that this fact has contributed to shape the behavior of both authors and journal editors accordingly.

On one hand, the use of journal-level bibliometric indicators has increasingly become a crucial element that authors use for choosing where to submit their manuscripts (Huggett, 2013). On the other hand, editors have started considering the improvement

of a journal impact factor an essential part of their editorial duty (Shanta, Pradhan, and Sharma, 2013). Editors may use several strategies, such as publishing an annual editorial referencing numerous articles published in the same journal in recent years or declining categories of publications that are unlikely to be cited (Huggett, 2013). Another strategy consists of requests asked by the editor during the reviewing process that make no suggestions to specific parts of a manuscript that need to be improved or papers that need to be cited, but only guide authors to add citations to papers in the journal handling the submission. This phenomenon of *coercitive citations* appears to occur in different scientific areas (Wilhite and Fong, 2012).

The relationships between actors of the publishing system w.r.t. the bibliometric indicators have been studied through models designed with the aim of capturing the mechanisms of the real academic system. With this concern in mind, Milojević, 2014 have presented a statistical model that showed the scientific research evolution through time, in which it was possible to investigate the difference in the growth of researchers groups in the different fields of science during the last decades. Other models of academic system have been made by Fortunato et al., 2018; Zeng et al., 2017, in which the authors have analyzed scientific and bibliometric data through network theory, in order to capture the trends in the evolution of science. These models could—as suggested by their authors—be used for improving the scientific enterprise and careers, for better performance evaluation of the productions, for discovering novel effective funding vehicles, and even for identifying promising research groups.

7.3 Model

7.3.1 Overview of the model

We model the academic publishing system as a discrete-time stochastic system where two kinds of *actors*, authors and editors, take actions in order to deal with two kinds of *resources*, papers and journals. Authors write papers and submit them to journals for publication. Editor are associated with journals and decide which papers submitted to their journal are to be published. The sets of authors, editors, and journals is defined statically.

We associate each paper with a numerical index that quantifies its scientific *quality*. We consider quality as an abstract notion, that is, we leave the features of a paper that influence its quality unspecified. We also associate each author with a numerical index, the author quality, that determines the quality of the papers produced by that author: the quality of a paper is drawn from a normal distribution parametrized by the author quality (for simplicity, we assume that each paper has a single author). We assume that author quality and paper quality do not change over time.

At each time step, each author can either submit the paper he/she is working on to a journal or write a new paper. Whenever a paper is submitted to a journal, the editor associated with that journal can either accept or reject the paper. Each paper is statically associated with (a) a number that quantifies its quality, and (b) a set of references, i.e., the papers cited by this paper. The papers in the set of references are selected at random, with the probability of selecting a paper proportional to the product of its quality and its recency.

In the following sections, we describe in details actors and resources involved in the model and how the system evolves over time. Moreover, we describe the bibliometric indexes that can be used to assess the state of the model.

7.3.2 Actors and resources

The system includes a set A of authors, a set P of papers, a set J of journals, and a set E of editors.

An *author* $a \in A$ is defined by its quality $q_A(a) \in [0, q_{\max}]$ and by its working paper $p_A(a) \in P$: the quality of an author do not change over time, whereas the working paper may change. The quality of an author determines the quality of the working paper, as specified in the next section.

A *paper* $p \in P$ has the following attributes: author $a_P(p) \in A \cup \{\varnothing\}$ (where $a_P(p) = \varnothing$ means that p has no authors); quality $q_P(p) \in [0, q_{\max}]$; set of references $C(p) \in \mathcal{P}(P_W)$, i.e., set of papers cited by p (where $P_W = \{p \in P : y(p) \neq \varnothing\}$ is the set of published papers) number of rejections $r_P(p) \in \mathbb{N}$; publication year $y(p) \in \mathbb{N} \cup \{\varnothing\}$ (where $y(p) = \varnothing$ means that p is not published). Quality of a paper, author, set of references do not change over time, while number of rejections and publication year may change over time.

A *journal* $j \in J$ has the following attributes: number of rejections $r_J(j) \in \mathbb{N}$; set of published papers $W(j) \in \mathcal{P}(P_W)$. Both attributes may change over time.

Finally, an *editor* $e \in E$ has a single attribute, that is the journal $j_E(e) \in J$ he/she is responsible for. This attribute does not change over time.

7.3.3 Bibliometric indexes

We consider three bibliometric indexes that are actually used for research assessment and apply them in our model.

For the assessment of authors, we consider the h-index (McDonald, 2005), computed for an author a as:

$$\mathrm{H}(a) = \max_{i \in \mathbb{N}} \min\left(|\{p \in P_W : C(p) \ni p_{a,i}\}|, i\right) \tag{7.1}$$

where $(p_{a,1}, p_{a,2}, \dots)$ is the sequence of papers published by a (i.e., such that $a_P(p) = a$ and $y(p) \neq \varnothing$) sorted by decreasing number of citations. We remark that in our model the h-index of an author can never decrease over time.

For the assessment of journals, we consider the impact factor and the acceptance rate. Given a journal j, its acceptance rate is computed as:

$$\mathrm{AR}(j) = \frac{|W(j)|}{r_J(j) + |W(j)|} \tag{7.2}$$

Given a journal j, its impact factor is computed as:

$$\mathrm{IF}(j) = \frac{\sum_{p \in P_0} |C(p) \cap P_{j,-1} \cap P_{j,-2}|}{|P_{j,-1}| + |P_{j,-2}|} \tag{7.3}$$

where $P_0 = \{p \in P : y(p) = y\}$ is the set of papers published at the current year y and $P_{j,-k} = \{p \in W(j) : y(p) = y - k\}$ is the set of papers published by journal j at year $y - k$. In other words, the impact factor is the ratio between the number of citations received this year by papers published by the journal in the last two years and the numbers of papers published by the journal in the last two years.

7.3.4 System evolution

The system starts at time $t = 0$ with sets A, J, E, P for authors, journals, editors, and papers, respectively. The initialization of these sets is described later. Then, the system evolves as follows, depending stochastically on the actions of authors and editors.

At each time step, each author has a *working paper* and takes an *action* based on a specified *author policy*. There are $1 + |J|$ possible actions, that may be "rewrite" (i.e., start working on a new paper) or "submit working paper to journal j" ("submit to j" in brief).

The "rewrite" action consists in removing the current working paper from P and creating a new working paper $p_A(a) \leftarrow p$. The attributes of p are set as follows: no publication year ($y(p) = \varnothing$); zero rejections ($r_P(p) = 0$); paper quality randomly set as $q_P(p) \sim N(q_A(a), \sigma)$, that is, sampled from the normal distribution parametrized with mean $q_A(a)$ (the author's quality) and variance σ, where σ is a model-wise parameter; set of references $C(p)$ composed of n_{cit} papers obtained by randomly sampling P_W with a probability of taking a paper p' proportional to the product of its quality $q_P(p')$ and recency $\frac{1}{\max(1,(y-y(p'))^3)}$, where $y = \lfloor \frac{t}{10} \rfloor$ is the current year—n_{cit} is a model-wise parameter.

The "submit to j" action triggers the immediate execution of an action by the editor of j. The actions available to an editor are either "accept" or "reject" and the editor selects the action based on a specified *editor policy*.

The "accept" action consists in publishing the corresponding paper p. That is, the paper publication is set to the current year ($y(p) \leftarrow y$), the journal set of published papers is updated by including p ($W(j) \leftarrow W(j) \cup \{p\}$). Execution of this action also provokes the creation of a new working paper for the corresponding author a.

The "reject" action consists in increasing the number of rejections of both the paper ($r_P(p) \leftarrow r_P(p) + 1$) and the journal ($r_J(j) \leftarrow r_J(j) + 1$).

We defined several different author policies and editor policies and analyzed the corresponding system evolution for a number of different combinations. We focused our analysis on the point of view of authors (evolution of h-index), of editors (evolution of journal impact factors), and of the publishing system as a whole (evolution of the overall quality of published papers). We considered only scenarios with the same policy for all the authors and the same policy for all the editors.

We considered two kinds of policies: *static* policies, that model simple yet representative behaviors of actors of the academic publishing system, and *learnable* policies, that are obtained by optimizing two template policies in order to achieve realistic goals of actors.

We defined all policies in such a way that authors are not aware of editor policies and vice versa. In other words, the behavior of authors is not influenced by the knowledge of how editors decide whether to accept a given submission. Similarly, editors are not aware of author policies, i.e., of the reason why an author has chosen to submit to a specific journal.

Static policies

As static policies for the authors, we considered the following two policies:

- *Random*. If the number of rejections of the working paper is greater than a specified system-wide threshold τ_r, then the action is "rewrite". Otherwise, the action is "submit to j" and j is chosen randomly with uniform probability in J.

- *Minded*. The action is always "submit to j" and j is chosen as follows. First, all journals are sorted according to their IF and grouped in n_{bin} bins, each bin containing $\left\lfloor \frac{|J|}{n_{\text{bin}}} \right\rfloor$ journals. Then, the k-th bin is selected based on the working paper quality and its number of rejections, as $k = \min\left(1, \left\lfloor \frac{q_P(p)}{q_{\max}} \right\rfloor - r_P(p)\right)$. Finally, the journal j is chosen randomly with uniform probability in the k-th bin.

The Random policy models the behavior of authors that are not interested in the bibliometric indexes associated with journals: authors keep on trying to have their current working paper published on some journal and give up after too many rejections. Furthermore, with this policy, authors do not take into account the quality of their papers in any way. The Minded policy instead models a scenario in which authors are aware of the quality of their papers and attempt to exploit this knowledge for publishing on journals with better bibliometric indexes. The choice of the journal where to submit a given paper is made by matching the paper quality to the journal IF, in terms of relative, discrete ranking.

As static policies for the editors, we considered the following three policies:

- *Accept-everything*. The action is always "accept".
- *Mild-scrutiny*. Given the paper p submitted to the journal j, the action is "accept" if $q_P(p) \geq \bar{q}_{25\,\%}(j)$ and "reject" otherwise, where $\bar{q}_{25\,\%}(j)$ is the first quartile of the quality of the papers $W(j)$ published by j.
- *Improve-only*. Given the paper p submitted to the journal j, the action is "accept" if $q_P(p) \geq \bar{q}_{50\,\%}(j)$ and "reject" otherwise, where $\bar{q}_{50\,\%}(j)$ is the median quality of the papers $W(j)$ published by j.

The Accept-everything policy does not take into account the quality of published papers in any way. The Mild-scrutiny policy models a behavior of editors that determine the quality of a submitted paper and accept a submission only if its quality is not exceedingly low with respect to the quality of the papers already published. Improve-only is similar to Mild-scrutiny, except that editors are only willing to publish papers that may improve the overall quality of the set of published papers.

Learnable policies

Concerning the learnable policies, we cast our model as a *reinforcemente learning (RL)* problem where the goal is to optimize a *template policy* for maximizing an objective function. The objective function for authors is the h-index while the objective function for editors is the impact factor. The template policy wraps a RL policy that takes an action in the set $\{\uparrow, \downarrow, \bigcirc\}$. The RL policy is described at the end of this section.

The template policy for authors is the same as the Minded static policy, except that the choice of the bin k of the journal ranking is based on the *ambition* $\alpha(a)$ of the author a, i.e., $k = \alpha(a)$. The ambition $\alpha(a) \in \{1, \ldots, n_{\text{bin}}\}$ is part of the state of the RL agent associated with the author a. The RL agent takes an action at each time step and $\alpha(a)$ is incremented, decremented, or kept unchanged depending on the action selected by the RL policy, $\uparrow$, $\downarrow$, and $\bigcirc$, respectively.

The template policy for editors is the same as the Mild-scrutiny policy, except that the quality threshold over the distributions of already published papers for deciding whether to accept a submission is not specified statically: it is instead based on the *selectivity* $\beta(e)$ of editor e. The selectivity $\beta(e) \in \{1, \ldots, n_{\text{bin}}\}$ is part of he state of the

RL agent associated with the editor e. The RL agent takes an action at each time step and $\beta(e)$ is incremented, decremented, or kept unchanged depending on the action selected by the RL policy, $\uparrow$, $\downarrow$, and $\bigcirc$, respectively—we remark that the editor may take zero or more actions at each time step, whereas its corresponding RL agent takes exactly one action at each time step.

Finally, the RL policy is obtained by means of Q-learning (Watkins and Dayan, 1992) and represented in a tabular form with different information available to authors and editors, as follows. For authors, the available knowledge is $\alpha(a), q_P(p_A(a)), r_P(p_A(a))$, i.e., current ambition, quality of the current working paper, and number of rejections of the current working paper. For editors, the available knowledge is $\beta(e), \mathrm{IF}(j), |W(j)|$, i.e., current selectivity, current impact factor of the journal, and current number of published papers.

We remark that the template policies are sufficiently expressive to model realistic and interesting behaviors. For example, authors might always keep a low ambition and submit their papers to journal with a low impact factor, regardless of the paper quality, or the opposite. Editors might increase or decrease selectivity based on the number of published papers, the impact factor, or both.

7.3.5 Model initialization

The model must be initialized with non-empty sets J, P, A (set E is implicitly specified by J). The size of these sets is a parameter. The content of P determines the initial IF of each element of J. We assume that, initially, each author has not published any paper. In order to initialize the model, thus, we need to select the quality of authors and, for each published paper, its quality, set of references and journal where the paper is published.

We assume that both the quality of authors and the quality of published papers is described by a power law distribution parameterized by (θ, k): intuitively, many authors and papers with low quality, few authors and papers with very high quality. The quality of authors and of papers are determined by sampling that distribution. For assigning a paper to a journal at initialization, we used a procedure aimed at obtaining, just after the initialization, a distribution of the (rounded) impact factors which resembles the real one, as measured in 2018 (see Figure 7.1). In particular, for each paper p being generated, we (1) compute the $|J|$ different IF distributions obtainable by adding p to each journal $j \in J$ and (2) actually add p to the journal for which the resulting distribution is the closest to the real IF distribution according to the Euclidean distance.

7.4 Experiments

We considered scenarios with $N_A = 100$ authors, $N_E = 20$ editors and $N_P^0 = 1000$ initial number of papers. The simulation length was $y_{\max} = 20$ years, with every year taking 10 time step. We analyzed 8 different scenarios corresponding to different combinations of policies for authors and editors, as follows.

- Initially, we executed a suite of 4 simulations, one for each combination of the two policies Random, Minded for authors with Improve-only, Mild-scrutiny for editors.

- Then, we executed 2 further simulations with policies based on reinforcement learning: RL editor with Minded author and Improve-only editor with RL

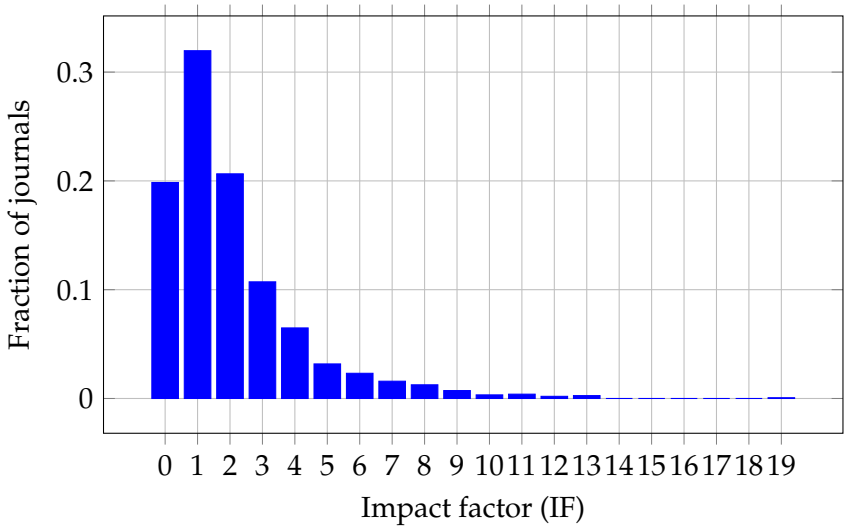


FIGURE 7.1: Impact factor distribution in 2018 according to Scimago.

TABLE 7.1: Predefined parameters

Parameter	Meaning	Value
N_A	Number of authors	100, 1000
N_E	Number of editors	20, 40
n_{ref}	Number of references in each paper	10
θ	Power law domain maximum	5
k	Power law slope	1.5
σ	Author variance	1
$\bar{Q}^P$	Maximum value for p-quality	15
N_P^0	Initial number of accepted papers	1000, 10000
k, u	Author intrinsic quality params.	
y_{max}	Simulation last year	20
q_{max}	Maximum paper quality	15
I_{max}	Maximum impact factor	20
n_{rejects}	Maximum number of rejects	10
n_{bins}	Number of bins	10

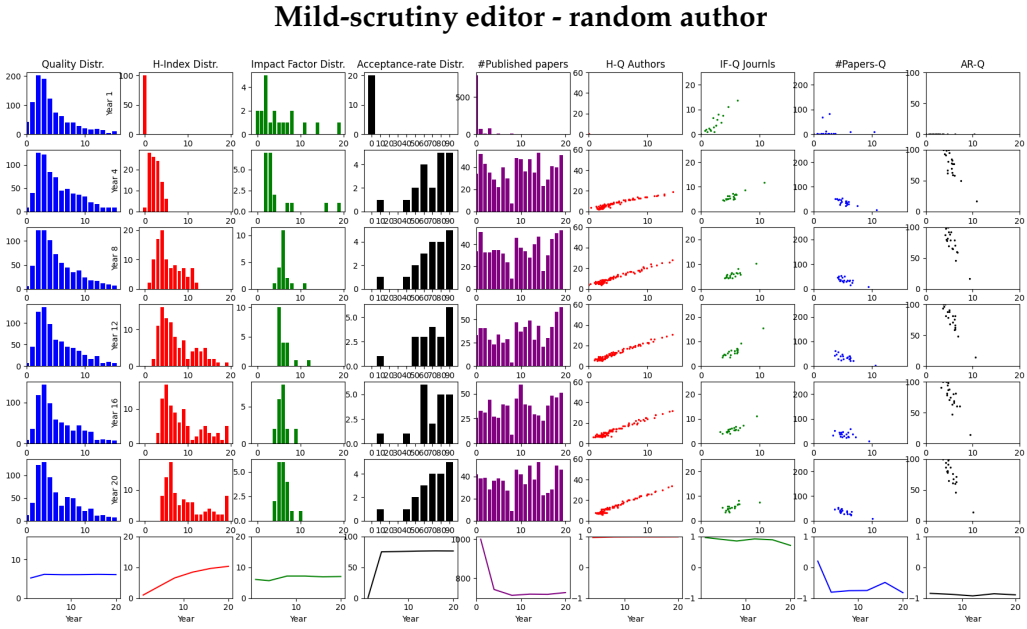


FIGURE 7.2: Experimental results with $N_E = 20$ editors employing the mild-scrutiny policy and $N_A = 100$ authors employing the random policy, and with $N_P^0 = 1000$ initially published papers.

author. We chose these two specific combinations after analyzing the results of the first set of simulations.

- Finally, we executed 2 further simulations: Improve-Only-Minded and Mild Scrutiny-Minded with greater parameter values: $N_A = 1000$ authors, $N_E = 40$ editors, $N_P^0 = 10000$ initial number of papers.

For each scenario, we analyzed a number of different indexes and their evolution across time. We describe each scenario by means of a figure containing a set of diagrams arranged as a grid, as follows (please refer to Figure 7.2). Each column corresponds to a specific analysis while the first five rows correspond to snapshots taken at years 1, 4, 8, 12, 15 and 20, respectively. In detail, the first four columns plot distributions: paper quality, author h-index, journal impact factor, journal acceptance rate; the fifth column plots the number of published papers by each journal; the remaining columns provide relations between pairs of indexes: h-index vs. author quality (one point for each author), impact factor vs. quality of published papers, number of published papers vs. quality for journals, acceptance rate vs. quality of published papers for journals. Finally, the last row provides the time evolution of the following indexes: average value of distribution across time, for the first five columns; correlation between the pair of indexes, for the remaining three columns.

7.5 Results

7.5.1 Empirical findings about the authors

A key finding is that there is a strong correlation between author quality and h-index in all the scenarios that we considered (column 6, H-Q authors in Figure 7.2 through Figure 7.9). Thus, according to our model, using h-index as a proxy for estimating author quality is justified. It is important to emphasize, though, that our model embeds the assumption that the choice of references is indeed biased by the quality

Mild-scrutiny editor - minded author

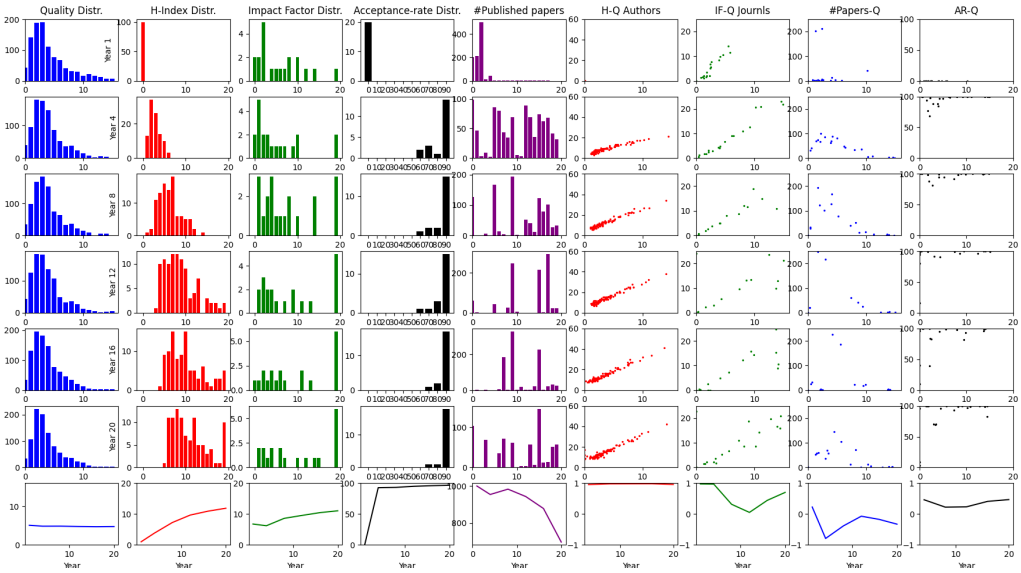


FIGURE 7.3: Experimental results with $N_E = 20$ editors employing the mild-scrutiny policy and $N_A = 100$ authors employing the minded policy, and with $N_P^0 = 1000$ initially published papers.

Improve-only editor - random author

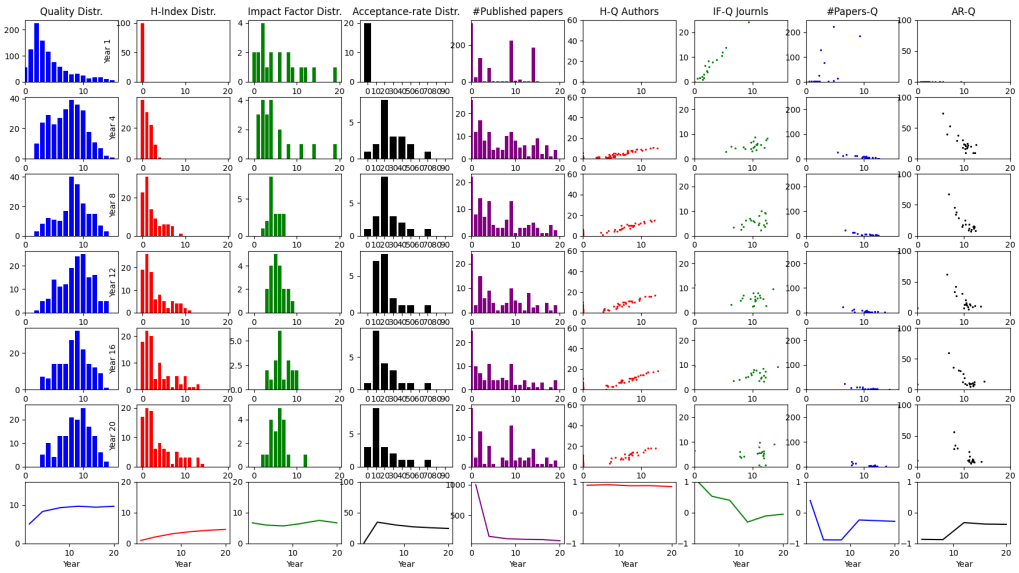


FIGURE 7.4: Experimental results with $N_E = 20$ editors employing the improve-only policy and $N_A = 100$ authors employing the random policy, and with $N_P^0 = 1000$ initially published papers.

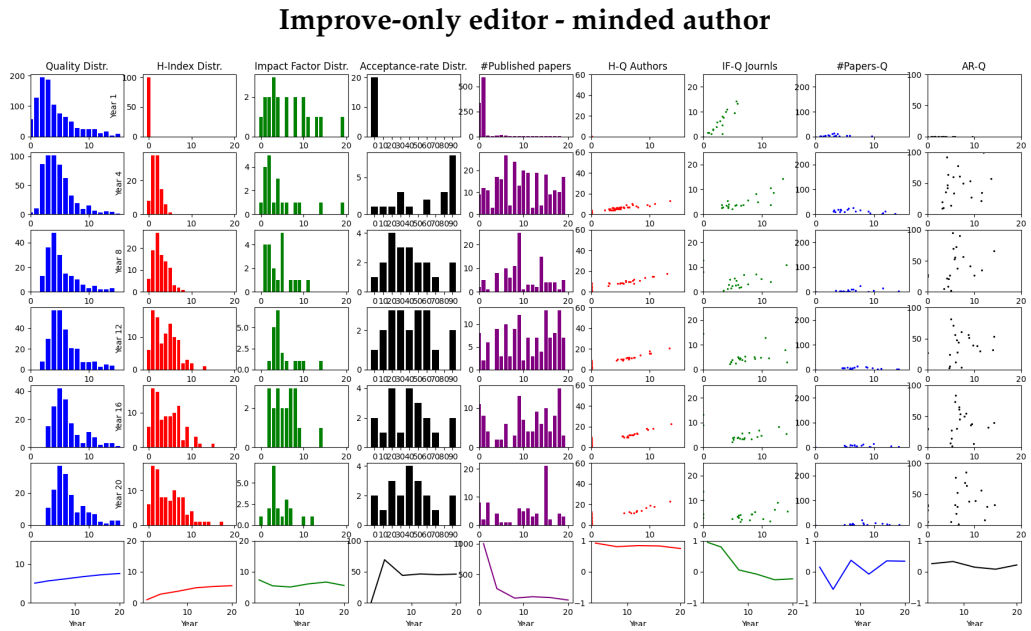


FIGURE 7.5: Experimental results with $N_E = 20$ editors employing the improve-only policy and $N_A = 100$ authors employing the minded policy, and with $N_P^0 = 1000$ initially published papers.

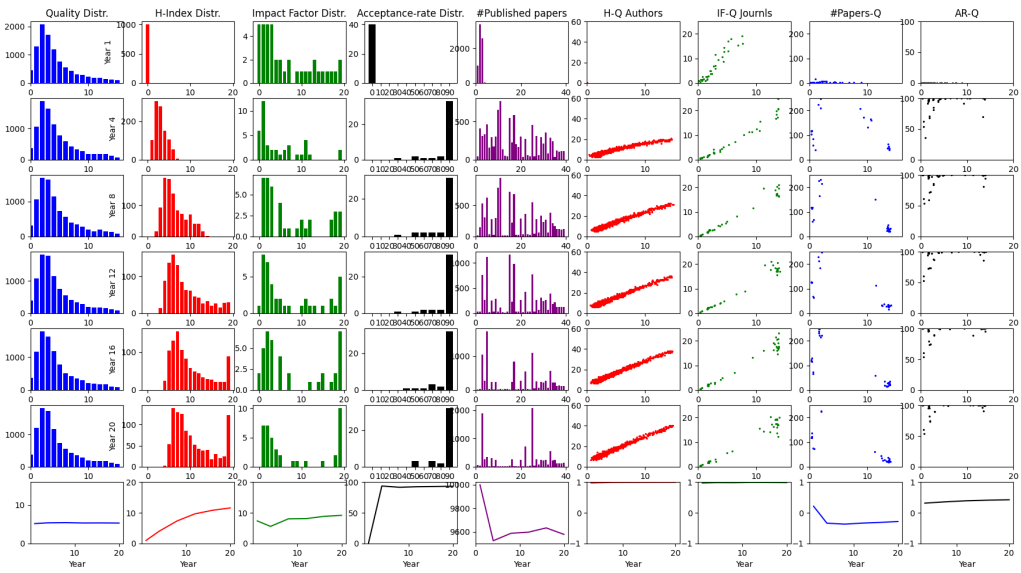


FIGURE 7.6: Experimental results with $N_E = 40$ editors employing the mild-scrutiny policy and $N_A = 1000$ authors employing the minded policy, and with $N_P^0 = 10000$ initially published papers.

Improve-only editor - Minded author [Big Exp]

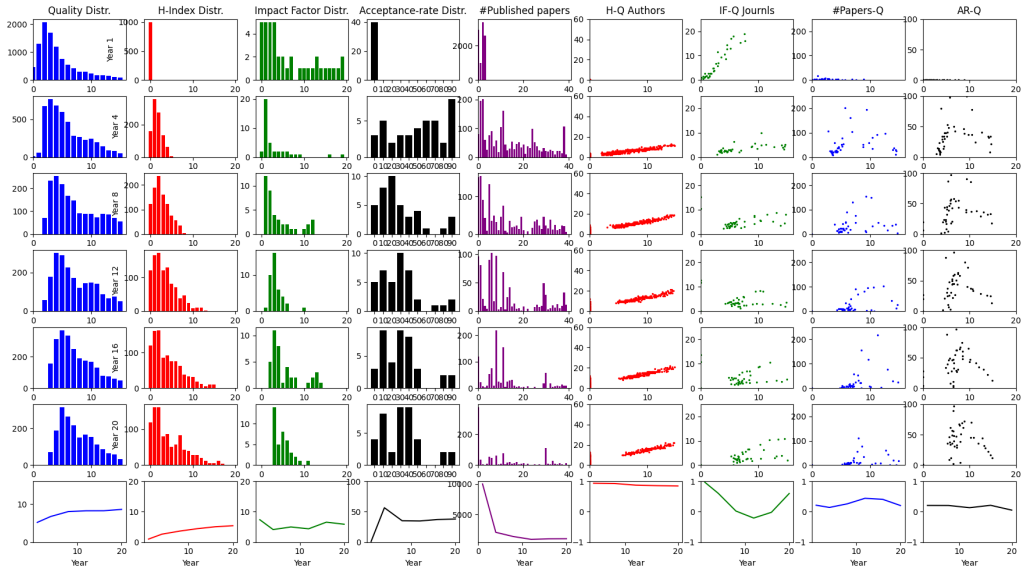
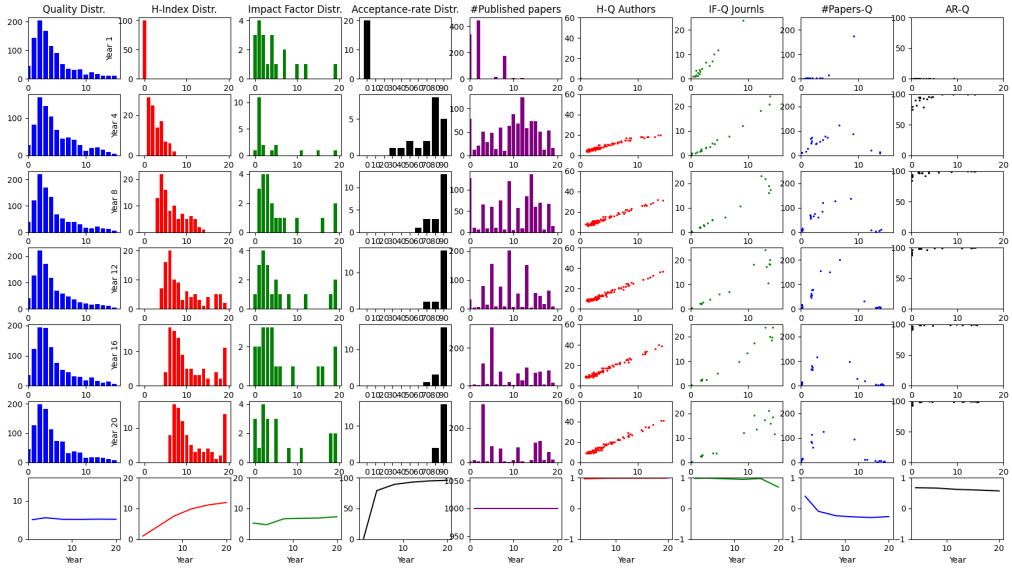


FIGURE 7.7: Experimental results with $N_E = 40$ editors employing the improve-only policy and $N_A = 1000$ authors employing the minded policy, and with $N_P^0 = 10000$ initially published papers.

RL editor - minded author

FIGURE 7.8: Experimental results with $N_E = 20$ editors employing the RL policy and $N_A = 100$ authors employing the minded policy, and with $N_P^0 = 1000$ initially published papers.



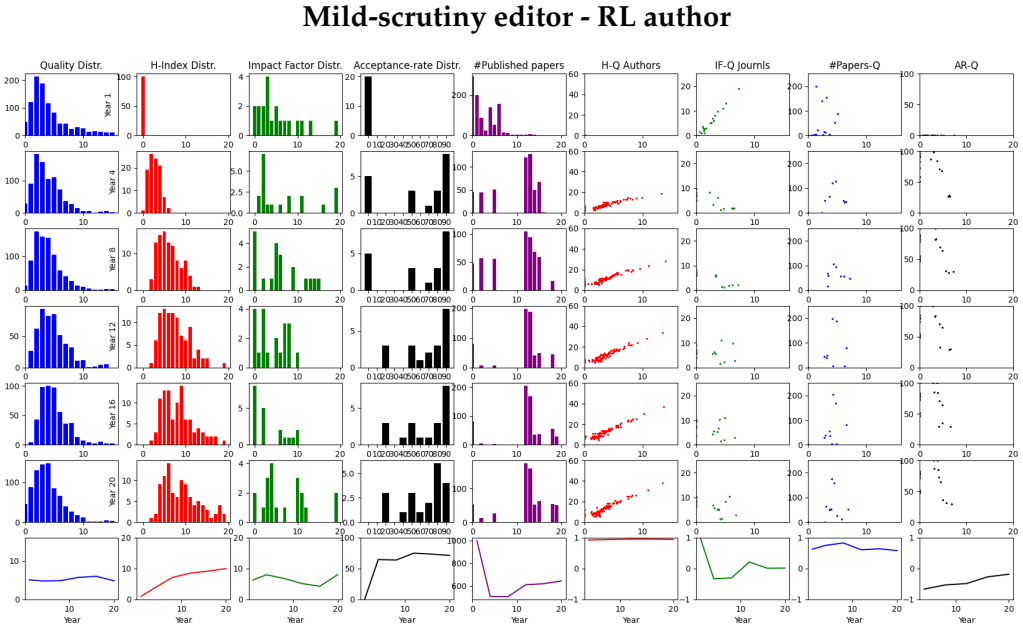


FIGURE 7.9: Experimental results with $N_E = 20$ editors employing the mild-scrutiny policy and $N_A = 100$ authors employing the RL policy, and with $N_P^0 = 1000$ initially published papers.

of references themselves. This assumption has been often criticized in the literature, on the grounds that the choice of references may be influenced by several additional factors that have little to do with the intrinsic quality of those references, such as, e.g., discoverability by search engines and research trends.

Investigating further the impact of the authors policy on the overall system evolution, in the scenarios in which the authors employ respectively the random and the minded policy, it seems that there are few major qualitative differences in the experimental results, that we briefly present here. The first main difference is that if the authors employ the random policy (Figure 7.2, and Figure 7.4), it holds that the resulting published papers distribution is uniform w.r.t. the journals. In other words each journal is as likely as the others to publish a paper, regardless of its impact factor, past published papers, and even regardless of its editor policy. Otherwise if the authors employ the minded policy (Figure 7.3, and Figure 7.5), this leads to a published papers distribution such that few journals are responsible for most of the published papers, while the majority of the journals publish few, or even no paper at all.

The second main difference is that when each author submits its papers to random journals, as it occurs in Figure 7.2 and Figure 7.4, it holds that the mean published papers quality is anti-correlated with the journals acceptance-rate. Each editor often ends up rejecting papers submitted to its journal, especially when applying a strict scrutiny like in Figure 7.4, thus decreasing its journal acceptance-rate, and then resulting in overall higher published papers quality. On the other hand, if the authors employ the minded policy, like in Figure 7.3 and Figure 7.5, each journal mean published papers quality is loosely correlated with its acceptance-rate w.r.t. the submissions received, and this correlation decreases as the editors employ a more strict policy, from Figure 7.2 to Figure 7.4.

7.5.2 Empirical findings about the editors

In this paragraph we consider the impact of the editors policy on the experiments outcome. To do so we compare the experimental results in which the editors employ different scrutiny severity, given the same policy employed by authors.

The main finding that we present here is that the evolution over time of the published papers quality distribution seems to be affected mostly by the editors policy, and more specifically depending on the scrutiny employed. In particular, for all the scenarios in which the editors employ a strict scrutiny (Figure 7.4, Figure 7.5, and Figure 7.7), the number of published papers decreases over time, and the published papers quality distribution changes from the initial power-law distribution, to look like a Gaussian distribution with mean value close to the center of the papers quality range $\frac{q_{\max}}{2}$. On the other hand, if the editors employ a not too strict scrutiny (Figure 7.2, Figure 7.3, Figure 7.6), it can be observed that the mean published papers quality of a journal is a good predictor of its impact-factor. This result suggests that the impact-factor of a journal is almost never a good proxy for its mean published papers quality, except in specific scenarios in which the editors employ a mild scrutiny policy.

7.5.3 Empirical findings about the learnable policies

In this last section we consider the outcome of the experiments in which the actors policies are optimized by means of RL. From all the results discussed so far it seems that there is no strong correlation between the number of papers published by a journal and its mean published papers quality. Surprisingly, the only scenario in which the number of papers published by a journal is highly correlated to the quality of that journal, is the one in which the authors employ the learnable policy, as showed in Figure 7.9.

Moreover it seems that H-index optimization and impact-factor optimization are conflicting objectives, at least as far as concern the correlation between the journals mean published papers quality and their impact-factor. Specifically, by looking at the results at the end of the training in Figure 7.9, where each author goal is to optimize its H-index, it seems that the published papers quality is not a good predictor of the journals impact-factor. This means that having the editors employing a not too strict scrutiny is a necessary—but not sufficient—condition for allowing the correlation between the mean published papers quality of a journal and its impact-factor. In other words, even if the editors employ a not too strict policy, and the authors aim at improving their H-index, the authors resulting behavior opposes to the correlation between journals impact-factor and their mean published papers quality. Otherwise if the editors employ the learnable policy, and the authors employ the minded policy, like in Figure 7.8, the correlation between the journals impact-factor and their mean published papers quality at the end of the training is high.

To sum up this last finding, it seems that when actors employ the learnable policies, the authors decrease the correlation between impact-factor and mean published quality, in order to improve their H-index, whereas the editors increase this correlation when aiming at improving their impact-factor.

7.6 Concluding remarks

A correct assessment of the scientific production is extremely important for the progress of knowledge, but it is at the same time a challenging problem. To this extent we have considered a model of the publishing system, in which the scientific

actors assessment is done using some measurable quantities, that act as proxies for other non-measurable ones. We have investigate the suitability of the currently used indicators, namely H-index and Impact factor, which are respectively used for measuring the quality of authors and journals. We have simulated the evolution of the publishing system, where the actors are provided with predefined strategies and learnable ones as well. The experimental results show that H-index is indeed a good indicator, since its fairly represent its author non-observable quality. On the other side, the Impact factor seems in general not as good as a predictor of the non-observable quality of its journal. However, if the editors employ a mild scrutiny on the submitted papers, the consequent journals Impact factor seems to be meaningful of the quality of those journals.

Chapter 8

Concluding Discussion

8.1 Research Questions

8.1.1 Answering Research Question 1

Which is the optimal language expressiveness for learning the most effective and efficient behaviors in a cooperative multi-agent RL?

To answer this research question, we have considered a cooperative multi-agent scenario, in which all the agents must learn to communicate in order to succeed at the game. Specifically, each agent must learn to recognize its own target from what it hears, to move toward that target, and finally to communicate useful information to guide the other agents.

In Chapter 2 we showed that the optimal language expressiveness is the one that can be obtained when the vocabulary size, that is the number of words that can be produced, is close to the product between number of agents and number of targets, which are all the entities within the game. In other words a vocabulary size smaller than this value defines a language that is not expressive enough for that problem complexity. On the other side, a vocabulary size bigger than this number allows too many words to be generated, thus making it difficult for the RL to explore the resulting larger search space, and thus negatively affecting the learned policies.

While these results present an approach for correctly designing a language that allows the agents to optimally learn to communicate in this simple game, there are a few limitations that we are aware of, and that we briefly list here:

- The form of communication defined in Chapter 2 allows only one-hot words to be uttered. Despite simplifying the communication learning, this design choice limits the number of words that can be expressed with a fixed vocabulary size. An interesting aspect that might be worth to investigate is the possibility to allow more words, by removing the constraint of them being one-hot, and then measuring the effectiveness and efficiency of the policies learned through RL with an increasingly limited vocabulary size.

- Another aspect that might limit the scope of these results is that the communication in Chapter 2 is broadcast by design. It might be interesting to extend these results with non-broadcast form of communications, where the agents able to hear are defined by the game, i.e. communication based on the agents position within the game, where agents have to be close enough to communicate, or this aspect might be treated as part of the optimization, i.e. including in the agents policy the possibility to choose who to direct their communication to.

8.1.2 Answering Research Question 2

What is the impact on the individual-collective reward relationship of learning the communication strategies within a Multi-Agent RL game?

This aspect has been investigated in Chapter 3, where we have experimentally assessed the role of communication in the emergence a of collective behaviour in multi-agent systems. In this game, each agent policy is learned by means of RL, and different scenarios w.r.t. the individual goal are given to the agents, where each scenario is characterized by a different relationship between individual and collective behavior. It is worth to notice also that in this work the agents policy learning occurs in a completely decentralized way, differently from what we did in Chapter 2, where only the validation was completely decentralized.

The experimental results presented in Chapter 3 show that the agents which learn to communicate, have developed a collective behaviour in the cooperative scenarios, and the resulting communication is indeed responsible for the equal allocation of resources among the agents, and the maximization of the individual goal as well. On the other side, the agents who learned to communicate in the competitive scenarios, developed a selfish behaviour and used their communication to promote inequality. This last result is motivated by the nature of the competitive scenario, since there is a conflict between the individual goal, which opposes to the equal allocation of resources, and the collective goal, which aims at minimizing the inequality.

Despite being extremely simple, this work provides some contributions to the understanding of multi-agent RL and towards the ethics of AI. However there are still many aspects worth to consider in a future work:

- In the work presented in Chapter 3 there is no form of regulation for assuring a minimum level of equality among the agents. It might be relevant in a future work to include a way to soften or punish the selfish tendency which arise in the competitive scenario, in order to discourage those behaviors, and investigate whether or not is possible to balance selfishness and equality in this way. This regulation might be enforced by means of a negative reward provided to the agents, similar to what we did in Chapter 4.

- Another aspect worth of considering, which is relevant to the ethics of AI discussed more in details in Chapter 4, is the possibility of letting the agents use communication to fake or to lie to others for their own profit. This aspect might allow even more complex selfish strategies in the competitive scenarios, but it requires to formalize the communication in a different way.

- Since in this work the focus was not on the development of the communication itself, but on the implications of learning to enable and disable the communication in a multi-agent system, the form of broadcasting employed in Chapter 3 is extremely simple. An aspect which has already been discussed in the section above, when we discussed about the limits of Chapter 2, is that also in Chapter 3 the communication is is very limited by design.

8.1.3 Answering Research Question 3

Is it possible to provide regulations for autonomous vehicles without explicitly disabling unethical behaviors?

To provide an answer to this question, in Chapter 4 we have designed a road traffic

model in which we have analyzed the system-wide indexes, namely efficiency and safety. In this scenario we have trained RL agents by providing them a form of traffic regulation inspired by the real traffic rules, while learning the optimal policy in this simulated system. The rules enforcement has been realized by providing a negative reward to the drivers that were not compliant with the regulation, in order to discourage those behaviors in the future. We have then experimentally measured the impact of these rules, by comparing this learning approach with another variant in which rules compliance does not impact on the reward function.

According to the experimental results, it is possible to improve the traffic safety by imposing these regulation during the RL agents policy learning, even if the increase in safety is obtained at the expense of the overall efficiency. This approach for enforcing the rules allows the drivers to learn whether to evade a certain rule or not, based on the current situation, and therefore this aspect is also useful to tackle the ethical problems specifics to the field of self-driving cars regulation. The most interesting aspect of this approach is therefore that no action is prohibited by design, but instead the rules that we provide are flexible, and the only principle guides the RL agents compliance to them is the maximization of the expected future return. In other words, a driver might have to overtake another vehicle in a situation in which overtaking is punished by the rules by a negative immediate reward, if this decision is the only one that allows avoiding a forthcoming collision, which in terms of reward corresponds to a higher penalty.

This work main contribution is therefore this flexible form of autonomous vehicles regulation, which can be extended in future works:

- The road traffic scenario considered in Chapter 4 is a very simple one, in which vehicles are all alike, and where the only drivers are the RL agents. We might investigate the robustness of the policies learned by the RL agents with these rules, while sharing the environment drivers of a different kind, i.e. human drivers or deterministic driving algorithm, who are not subjected to the same rules or perform risky actions. It would be interesting to assess how the driving policies learned with the approach presented in this study operate in such situations.

- It might be relevant to focus more deeply on this regulation approach, by considering a scenario tailored to clearly exhibit the trade-off between compliance with the rules and a greater good, and in which it would be possible to assess the effectiveness of the form of regulation employed.

8.1.4 Answering Research Question 4

Is it possible to take advantage of the body complexity by evolving a sensing neural controller for voxel-based soft robots?

We have provided an answer to this question in Chapter 5, where a form of control based on an artificial neural network, equipped with sensing capabilities distributed across the full body of the robot, has been proposed as an alternative to the non-sensing counterpart from the literature.

Given a fixed body and a task as a benchmark, we have synthesizing the optimal control algorithm by means of EC, for the sensing neural controller and the non-sensing one as well. In order to compare the performance of the two controllers, we have manually designed three different bodies and two different environments, in which we have evaluated the robots on a locomotion task. From the results of this

comparison, we have found that the sensing controller is generally more effective than its non-sensing counterpart, also when the robot is evaluated in a validation environment which is different from the one in which it has been evolved.

The idea of shifting the complexity from the controller to the body is one of the key elements that has motivated the development of voxel-based soft robots. However, based on the very promising results described in Chapter 5, we suggest that in order to successfully take advantage of the complexity of the body, these robots should be provided with a form of distributed sensing. We conclude by providing some ideas for future works:

- In Chapter 5 we have considered only simple bodies that could be manually designed, and such that their optimization required a relatively limited computational effort. Besides a set of benchmark tasks for voxel-based soft robots is currently missing, being most of the works in the literature based on the locomotion. As a future work, we plan to investigate the potential of sensing controllers on larger robots and more complex tasks.

- Considering bigger robots alone would result in a higher complexity algorithm for searching within the solutions space, which might not be efficiently done by the standard evolutionary algorithm used in this work. Therefore we plan to cope with this increased complexity by relying on a more efficient evolutionary framework, as well as a modular design for the robot bodies.

- The experimental results described in Chapter 5 show that another aspect worth to investigate might be the extension of the distributed sensing paradigm through the implementation of a distributed sensing controller. Such controller might allow to take even more advantage of the complexity of the body, and might improve the robot modularity as well.

8.1.5 Answering Research Question 5

Is it possible to automatically design a body for voxel-based soft robots that can adapt to differnt tasks?

To answer this question, in Chapter 6 we have provided a definition of adaptability, as the ability to perform fairly good on tasks requiring different skills. Consequently we we have defined a way to measure the adaptability of a body, based on the concept of self-organized criticality. We have optimized a number of soft-robots bodies towards adaptability by means of EC, and we have manually designed several other bodies, some of them inspired by the ones used in Chapter 5, while some others from the literature, and finally some others automatically generated using algorithm based on randomness. We have considered three tasks which require different motor skills, and we have optimized the same control variants defined in Chapter 5 for each body considered. The experimental results show that this approach based on the propery of self-organized criticality produces bodies that are generally more adaptable than the others. Specifically this bodies are more versatile than manually engineered bodies. The evolution towards adaptability is however a resources intensive process, therefore we show that it is possible to generate comparably adaptable bodies by means of algorithms involving randomness, if a sine wave is employed as control algorithm. However it seems also that when a sensing controller is employed, the bodies evolved towards self-organized criticality are able to take advantage of the distributed sensing to perform better than any other approach.

This result thus demonstrate that self-organized criticality is a factor responsible for granting adaptability to bodies, and it provides a guideline for the development of versatile bodies for voxel-based soft-robots. We plan to further explore this promising approach, in particular concerning the following research aspects:

- In this work three different tasks have been proposed to evaluate the adaptability. The choice of the tasks in Chapter 6 is arbitrary and based on qualitative observation of the robots behavior that has been observed when optimizing the controllers. Future works might consider a metric for the assessment of the difference between behaviors, and use it to design tasks that actually are proved to require different skills, in order to ensure a fair assessment of the adaptability.
- The adaptability assessment for a body in chapter 6 is based on fitting an empirical distribution which sample size depends on the number of voxels that body is made of, with a theoretical distribution obtained through linear regression. The correctness of the adaptability assessment is higher with a bigger sample size available. It might be worth to investigate the impact of the distribution resolution on the self-criticality assessment, and on the actual adaptability of a certain body. To investigate this aspect we might repeat the experiments done in Chapter 6, by considering bodies made of a higher number of voxels.

8.1.6 Answering Research Question 6

Are the currently employed publishing system indicators effective proxies for the corresponding non-observable quantities they embody?

An answer to this question, based on a simulated model of the publishing system, has been presented in Chapter 8. According to the model presented in Chapter 8, the authors and editors are modeled by artificial agents that are evaluated using the indexes currently adopted by the publishing system, which are respectively the H-index for the authors, and the Impact factor for the journals. These indicators stand for non-measurable quantities such as the quality of an author scientific production, and the quality of the papers published to a journal. In this simulated system we have defined many different policies for both the authors and the editors as well. By simulating the evolution of this dynamical system we have investigated the suitability of the indexes w.r.t. the agents policy. According to the experimental results, it seems that the H-index is a good approximation of the non-observable quality of the authors. The results on the Impact factor instead suggest that, if the editors employ a mild scrutiny, which is not too strict, but not even too loose, the Impact factor behaves as a good proxy for the journals non-observable quality. Therefore we have observed that in general the Impact factor is not a good index for the journals assessment, but instead it is a fair predictor only under a very specific condition.

The results showed in Chapter 8 suggest that the tool of multi-agent system is extremely versatile, and enables to study very complex phenomena, such as the dynamics of the publishing system. Despite the promising results of the proposed approach, we might consider some improvements:

- For instance we might focus on the development of unethical behaviors, motivated by the actors goal of maximizing their own indicator. This aspect is indeed very relevant, since the practice of unethical behaviors is a real problem

that affects the publishing system. By focusing on this aspect we might have a better understanding of what causes these behaviors, and in which way we can discourage them.

Bibliography

Agnoli, Franca et al. (2017). "Questionable research practices among Italian research psychologists". In: *PloS one* 12.3, e0172792.

Ahilan, Sanjeevan and Peter Dayan (2019). "Feudal multi-agent hierarchies for cooperative reinforcement learning". In: *arXiv preprint arXiv:1901.08492*.

Amin, Mayur and Michael Mabe (2004). "Impact factors use and abuse". In.

Andrés, Ana (2009). *Measuring academic research: How to undertake a bibliometric study*. Elsevier.

Arbanas, Barbara et al. (2016). "Aerial-ground robotic system for autonomous delivery tasks". In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, pp. 5463–5468.

Baccini, Alberto, Giuseppe De Nicolao, and Eugenio Petrovich (2019). "Citation gaming induced by bibliometric evaluation: A country-level comparative analysis". In: *PLoS One* 14.9, e0221212.

Bachwerk, Martin and Carl Vogel (2011). "Establishing linguistic conventions in task-oriented primeval dialogue". In: *Analysis of Verbal and Nonverbal Communication and Enactment. The Processing Issues*. Springer, pp. 48–55.

— (2012). "Language and friendships: A co-evolution model of social and linguistic conventions". In: *The Evolution Of Language*. World Scientific, pp. 34–41.

Bahdanau, D. et al. (June 2018). "Learning to Follow Language Instructions with Adversarial Reward Induction". In: *ArXiv e-prints*. arXiv: 1806.01946 [cs.AI].

Bak, Per, Chao Tang, and Kurt Wiesenfeld (1988). "Self-organized criticality". In: *Physical review A* 38.1, p. 364.

Bansal, Trapit et al. (2017). "Emergent complexity via multi-agent competition". In: *arXiv preprint arXiv:1710.03748*.

Barnes, Cameron (2017). "The h-index debate: an introduction for librarians". In: *The Journal of Academic Librarianship* 43.6, pp. 487–494.

Bartocci, Ezio et al. (2017). "Monitoring mobile and spatially distributed cyber-physical systems". In: *Proceedings of the 15th ACM-IEEE International Conference on Formal Methods and Models for System Design*, pp. 146–155.

Bartoli, Alberto and Eric Medvet (2014). "Bibliometric evaluation of researchers in the internet age". In: *The Information Society* 30.5, pp. 349–354.

Bartoli, Alberto et al. (2016). "Your paper has been accepted, rejected, or whatever: Automatic generation of scientific paper reviews". In: *International Conference on Availability, Reliability, and Security*. Springer, pp. 19–28.

Bertschinger, Nils and Thomas Natschläger (2004). "Real-time computation at the edge of chaos in recurrent neural networks". In: *Neural computation* 16.7, pp. 1413–1436.

Bojarski, Mariusz et al. (2016). "End to end learning for self-driving cars". In: *arXiv preprint arXiv:1604.07316*.

Bongard, Josh et al. (2016). "Material properties affect evolutions ability to exploit morphological computation in growing soft-bodied creatures". In: *Artificial Life Conference Proceedings 13*. MIT Press, pp. 234–241.

Bornmann, Lutz and Rüdiger Mutz (2015). "Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references". In: *Journal of the Association for Information Science and Technology* 66.11, pp. 2215–2222.

Bouton, Maxime et al. (2019). "Safe reinforcement learning with scene decomposition for navigating complex urban environments". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 1469–1476.

Braganza, David et al. (2007). "A neural network controller for continuum robots". In: *IEEE transactions on robotics* 23.6, pp. 1270–1277.

Brodeur, Simon and Jean Rouat (2012). "Regulation toward self-organized criticality in a recurrent spiking neural reservoir". In: *International Conference on Artificial Neural Networks*. Springer, pp. 547–554.

Brodsky, Jessica S (2016). "Autonomous vehicle regulation: How an uncertain legal landscape may hit the brakes on self-driving cars". In: *Berkeley Technology Law Journal* 31.2, pp. 851–878.

Bruder, Daniel et al. (2019). "Modeling and Control of Soft Robots Using the Koopman Operator and Model Predictive Control". In: *arXiv preprint arXiv:1902.02827*.

Calvaresi, Davide et al. (2016). "A framework based on real-time os and multi-agents for intelligent autonomous robot competitions". In: *2016 11th IEEE symposium on industrial embedded systems (SIES)*. IEEE, pp. 1–10.

Cao, Yongcan et al. (2012). "An overview of recent progress in the study of distributed multi-agent coordination". In: *IEEE Transactions on Industrial informatics* 9.1, pp. 427–438.

Capasso, Alessandro Paolo, Giulio Bacchiani, and Daniele Molinari (2020). "Intelligent Roundabout Insertion using Deep Reinforcement Learning". In: *arXiv preprint arXiv:2001.00786*.

Cheney, Nicholas, Jeff Clune, and Hod Lipson (2014). "Evolved electrophysiological soft robots". In: *Artificial Life Conference Proceedings 14*. MIT Press, pp. 222–229.

Cheney, Nick, Josh Bongard, and Hod Lipson (2015). "Evolving soft robots in tight spaces". In: *Proceedings of the 2015 annual conference on Genetic and Evolutionary Computation*. ACM, pp. 935–942.

Cheney, Nick et al. (2013). "Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding". In: *Proceedings of the 15th annual conference on Genetic and evolutionary computation*. ACM, pp. 167–174.

Christiansen, Morten H and Simon Kirby (2003). *Language evolution*. OUP Oxford.

Clauset, Aaron, Cosma Rohilla Shalizi, and Mark EJ Newman (2009). "Power-law distributions in empirical data". In: *SIAM review* 51.4, pp. 661–703.

Corrall, Sheila, Mary Anne Kennan, and Waseem Afzal (2013). "Bibliometrics and research data management services: Emerging trends in library support for research". In: *Library trends* 61.3, pp. 636–674.

Corucci, Francesco et al. (2018). "Evolving soft locomotion in aquatic and terrestrial environments: effects of material properties and environmental transitions". In: *Soft robotics* 5.4, pp. 475–495.

Cox, Trevor F and Michael AA Cox (2000). *Multidimensional scaling*. Chapman and hall/CRC.

Devlin, Sam and Daniel Kudenko (2016). "Plan-based reward shaping for multi-agent reinforcement learning". In: *The Knowledge Engineering Review* 31.1, pp. 44–58.

Drew, John H, Andrew G Glen, and Lawrence M Leemis (2000). "Computing the cumulative distribution function of the Kolmogorov–Smirnov statistic". In: *Computational statistics & data analysis* 34.1, pp. 1–15.

Duan, Yan et al. (2016). "Benchmarking deep reinforcement learning for continuous control". In: *International Conference on Machine Learning*, pp. 1329–1338.

Elkind, Edith, G Chalkiadakis, and MJ Wooldridge (2012). *Computational aspects of cooperative game theory*.

Ellegaard, Ole and Johan A Wallin (2015). "The bibliometric analysis of scholarly production: How great is the impact?" In: *Scientometrics* 105.3, pp. 1809–1831.

Fernando, Chrisantha and Sampsa Sojakka (2003). "Pattern recognition in a bucket". In: *European conference on artificial life*. Springer, pp. 588–597.

Foerster, Jakob et al. (2016). "Learning to communicate with deep multi-agent reinforcement learning". In: *Advances in Neural Information Processing Systems*, pp. 2137–2145.

Fortunato, Santo et al. (2018). "Science of science". In: *Science* 359.6379.

Garfield, Eugene (1972). "Citation analysis as a tool in journal evaluation". In: *Science* 178.4060, pp. 471–479.

Gibbons, Thomas E (2010). "Unifying quality metrics for reservoir networks". In: *The 2010 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 1–7.

Glorot, Xavier and Yoshua Bengio (2010). "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256.

Grigorescu, Sorin et al. (2019). "A survey of deep learning techniques for autonomous driving". In: *Journal of Field Robotics*.

Hansen, Nikolaus, Sibylle D Müller, and Petros Koumoutsakos (2003). "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)". In: *Evolutionary computation* 11.1, pp. 1–18.

Havrylov, Serhii and Ivan Titov (2017). "Emergence of language with multi-agent games: learning to communicate with sequences of symbols". In: *Advances in Neural Information Processing Systems*, pp. 2146–2156.

Heiney, Kristine et al. (2019). "Assessment and manipulation of the computational capacity of in vitro neuronal networks through criticality in neuronal avalanches". In: *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, pp. 247–254.

Herron, Jennifer (2017). "Predatory publishers". In: *Journal of Electronic Resources in Medical Libraries* 14.1, pp. 27–31.

Hicks, Diana et al. (2015). "Bibliometrics: the Leiden Manifesto for research metrics". In: *Nature* 520.7548, pp. 429–431.

Hiller, Jonathan and Hod Lipson (2011). "Automatic design and manufacture of soft robots". In: *IEEE Transactions on Robotics* 28.2, pp. 457–466.

— (2012). "Automatic design and manufacture of soft robots". In: *IEEE Transactions on Robotics* 28.2, pp. 457–466.

Hirsch, Jorge E (2005). "An index to quantify an individual's scientific research output". In: *Proceedings of the National academy of Sciences* 102.46, pp. 16569–16572.

Hoel, Carl-Johan, Krister Wolff, and Leo Laine (2018). "Automated speed and lane change decision making using deep reinforcement learning". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 2148–2155.

Holstein, Tobias, Gordana Dodig-Crnkovic, and Patrizio Pelliccione (2018). "Ethical and social aspects of self-driving cars". In: *arXiv preprint arXiv:1802.04103*.

Howard, Daniel and Danielle Dai (2014). "Public perceptions of self-driving cars: The case of Berkeley, California". In: *Transportation research board 93rd annual meeting*. Vol. 14. 4502, pp. 1–16.

Huggett, Sarah (2013). "Journal bibliometrics indicators and citation ethics: a discussion of current issues". In: *Atherosclerosis* 230.2, pp. 275–277.

Hughes, Edward et al. (2018). "Inequity aversion improves cooperation in intertemporal social dilemmas". In: *Advances in neural information processing systems*, pp. 3326–3336.

Iida, Fumiya and Cecilia Laschi (2011). "Soft robotics: challenges and perspectives". In: *Procedia Computer Science* 7, pp. 99–102.

Isele, David et al. (2017). "Navigating intersections with autonomous vehicles using deep reinforcement learning". In: *arXiv preprint arXiv:1705.01196*.

Jaeger, Herbert (2002). "Adaptive nonlinear system identification with echo state networks". In: *Advances in neural information processing systems* 15, pp. 609–616.

Jang, Eric, Shixiang Gu, and Ben Poole (2016). "Categorical reparameterization with gumbel-softmax". In: *arXiv preprint arXiv:1611.01144*.

Jaques, Natasha et al. (2019). "Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning". In: *International Conference on Machine Learning*, pp. 3040–3049.

Jaritz, Maximilian et al. (2018). "End-to-end race driving with deep reinforcement learning". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2070–2075.

Johnson, Emmanuel, Jonathan Gratch, and David DeVault (2017). "Towards An Autonomous Agent that Provides Automated Feedback on Students' Negotiation Skills". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, pp. 410–418.

Kim, Sangbae, Cecilia Laschi, and Barry Trimmer (2013). "Soft robotics: a bioinspired evolution in robotics". In: *Trends in biotechnology* 31.5, pp. 287–294.

Kiran, B Ravi et al. (2020). "Deep Reinforcement Learning for Autonomous Driving: A Survey". In: *arXiv preprint arXiv:2002.00444*.

Kirby, Simon, Tom Griffiths, and Kenny Smith (2014). "Iterated learning and the evolution of language". In: *Current opinion in neurobiology* 28, pp. 108–114.

Kirkpatrick, Keith (July 2015). "The Moral Challenges of Driverless Cars". In: *Commun. ACM* 58.8, pp. 19–20. ISSN: 0001-0782. DOI: 10.1145/2788477. URL: http://doi.acm.org/10.1145/2788477.

Kriegman, Sam, Nick Cheney, and Josh Bongard (2018). "How morphological development can guide evolution". In: *Scientific reports* 8.1, p. 13934.

Kriegman, Sam et al. (2017). "Simulating the evolution of soft and rigid-body robots". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, pp. 1117–1120.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1097–1105.

Langton, Christopher (1990). *Computation at the edge of chaos: Phase transition and emergent computation*. Tech. rep. Los Alamos National Lab., NM (USA).

Lehman, Joel et al. (2018). "The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities". In: *arXiv preprint arXiv:1803.03453*.

Leibo, Joel Z et al. (2017). "Multi-agent reinforcement learning in sequential social dilemmas". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, pp. 464–473.

Lewis, Mike et al. (2017). "Deal or no deal? end-to-end learning for negotiation dialogues". In: *arXiv preprint arXiv:1706.05125*.

Li, Tao et al. (2012). "Behavior switching using reservoir computing for a soft robotic arm". In: *2012 IEEE International Conference on Robotics and Automation*. IEEE, pp. 4918–4924.

Liebner, Martin et al. (2012). "Driver intent inference at urban intersections using the intelligent driver model". In: *2012 IEEE Intelligent Vehicles Symposium*. IEEE, pp. 1162–1167.

Lin, Huai-Ti, Gary G Leisk, and Barry Trimmer (2011). "GoQBot: a caterpillar-inspired soft-bodied rolling robot". In: *Bioinspiration & biomimetics* 6.2, p. 026007.

Lipson, H. (2014). "Challenges and Opportunities for Design, Simulation, and Fabrication of Soft Robots". In.

Littman, Michael L (1994). "Markov games as a framework for multi-agent reinforcement learning". In: *Machine Learning Proceedings 1994*. Elsevier, pp. 157–163.

Loiacono, Daniele et al. (2010). "Learning to overtake in TORCS using simple reinforcement learning". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–8.

Lombardi, Giuseppe, Eric Medvet, and Alberto Bartoli (2017). "A Language for UAV Traffic Rules in an Urban Environment and Decentralized Scenario". In: *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, pp. 1139–1143.

Lowe, Ryan et al. (2017). "Multi-agent actor-critic for mixed cooperative-competitive environments". In: *Advances in Neural Information Processing Systems*, pp. 6382–6393.

Maaten, Laurens van der and Geoffrey Hinton (2008). "Visualizing data using t-SNE". In: *Journal of machine learning research* 9.Nov, pp. 2579–2605.

Maqueda, Ana I et al. (2018). "Event-based vision meets deep learning on steering prediction for self-driving cars". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5419–5427.

Mazzolai, B et al. (2012). "Soft-robotic arm inspired by the octopus: II. From artificial requirements to innovative technological solutions". In: *Bioinspiration & biomimetics* 7.2, p. 025005.

Mazzolini, Andrea and Antonio Celani (2020). "Generosity, selfishness and exploitation as optimal greedy strategies for resource sharing". In: *Journal of Theoretical Biology* 485, p. 110041.

McDonald, Kim (2005). "Physicist proposes new way to rank scientific output". In: *Phys Org*.

McPartland, Michelle, Stefano Nolfi, and Hussein A Abbass (2005). "Emergence of communication in competitive multi-agent systems: a pareto multi-objective approach". In: *Proceedings of the 7th annual conference on Genetic and evolutionary computation*. ACM, pp. 51–58.

Medvet, Eric, Alberto Bartoli, and Jacopo Talamini (2017). "Road traffic rules synthesis using grammatical evolution". In: *European Conference on the Applications of Evolutionary Computation*. Springer, pp. 173–188.

Medvet, Eric et al. (2020a). "2D-VSR-Sim: A simulation tool for the optimization of 2-D voxel-based soft robots". In: *SoftwareX* 12, p. 100573.

— (2020b). "Design, Validation, and Case Studies of 2D-VSR-Sim, an Optimization-friendly Simulator of 2-D Voxel-based Soft Robots". In: *arXiv*, arXiv–2001.

Milojević, Staša (2014). "Principles of scientific research team formation and evolution". In: *Proceedings of the National Academy of Sciences* 111.11, pp. 3984–3989.

Mirchevska, Branka et al. (2018). "High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 2156–2162.

Mnih, Volodymyr et al. (2013). "Playing atari with deep reinforcement learning". In: *arXiv preprint arXiv:1312.5602*.

Mnih, Volodymyr et al. (2015). "Human-level control through deep reinforcement learning". In: *Nature* 518.7540, p. 529.

Mnih, Volodymyr et al. (2016). "Asynchronous methods for deep reinforcement learning". In: *International conference on machine learning*, pp. 1928–1937.

Mordatch, Igor and Pieter Abbeel (2017). "Emergence of grounded compositional language in multi-agent populations". In: *arXiv preprint arXiv:1703.04908*.

— (2018). "Emergence of grounded compositional language in multi-agent populations". In: *Thirty-Second AAAI Conference on Artificial Intelligence*.

Nagelkerke, Nico JD et al. (1991). "A note on a general definition of the coefficient of determination". In: *Biometrika* 78.3, pp. 691–692.

Naghizadeh, Parinaz et al. (2019). "Hurts to Be Too Early: Benefits and Drawbacks of Communication in Multi-Agent Learning". In: *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. IEEE, pp. 622–630.

Nakajima, Kohei et al. (2013). "A soft body as a reservoir: case studies in a dynamic model of octopus-inspired soft robotic arm". In: *Frontiers in computational neuroscience* 7, p. 91.

Nenzi, Laura et al. (2015). "Qualitative and quantitative monitoring of spatio-temporal properties". In: *Runtime Verification*. Springer, pp. 21–37.

Nyholm, Sven and Jilles Smids (2018). "Automated cars meet human drivers: Responsible human-robot coordination and the ethics of mixed traffic". In: *Ethics and Information Technology*, pp. 1–10.

O'Neill, Michael and Conor Ryan (2001). "Grammatical evolution". In: *IEEE Transactions on Evolutionary Computation* 5.4, pp. 349–358.

Panait, Liviu and Sean Luke (2005). "Cooperative multi-agent learning: The state of the art". In: *Autonomous agents and multi-agent systems* 11.3, pp. 387–434.

Papoudakis, Georgios et al. (2019). "Dealing with Non-Stationarity in Multi-Agent Deep Reinforcement Learning". In: *arXiv preprint arXiv:1906.04737*.

Peroni, Silvio et al. (2020). "The practice of self-citations: a longitudinal study". In: *Scientometrics*, pp. 1–30.

Pfeifer, Rolf and Josh Bongard (2006). *How the body shapes the way we think: a new view of intelligence*. MIT press.

Pontes-Filho, Sidney et al. (2020). "A neuro-inspired general framework for the evolution of stochastic dynamical systems: Cellular automata, random Boolean networks and echo state networks towards criticality". In: *Cognitive Neurodynamics*, pp. 1–18.

Qiao, Zhiqian et al. (2018). "Pomdp and hierarchical options mdp with continuous actions for autonomous driving at intersections". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 2377–2382.

Rahwan, Iyad et al. (2019). "Machine behaviour". In: *Nature* 568.7753, p. 477.

Rahwan, Talal et al. (2015). "Coalition structure generation: A survey". In: *Artificial Intelligence* 229, pp. 139–174.

Ren, Wei, Randal W Beard, and Ella M Atkins (2007). "Information consensus in multivehicle cooperative control". In: *IEEE Control systems magazine* 27.2, pp. 71–82.

Rizaldi, Albert and Matthias Althoff (2015). "Formalising traffic rules for accountability of autonomous vehicles". In: *2015 IEEE 18th International Conference on Intelligent Transportation Systems*. IEEE, pp. 1658–1665.

Rossi, Federico et al. (2018). "Review of multi-agent algorithms for collective behavior: a structural taxonomy". In: *arXiv preprint arXiv:1803.05464*.

Ruocco, G. et al. (2017). "Bibliometric indicators: the origin of their log-normal distribution and why they are not a reliable proxy for an individual scholar's talent". In: *Palgrave Communications* 3, p. 17064.

Rus, Daniela and Michael T Tolley (2015). "Design, fabrication and control of soft robots". In: *Nature* 521.7553, pp. 467–475.

Sadeghi, Ali, Alessio Mondini, and Barbara Mazzolai (2017). "Toward self-growing soft robots inspired by plant roots and based on additive manufacturing technologies". In: *Soft robotics* 4.3, pp. 211–223.

Sallab, Ahmad EL et al. (2017). "Deep reinforcement learning framework for autonomous driving". In: *Electronic Imaging* 2017.19, pp. 70–76.

Saunders, William et al. (2018). "Trial without error: Towards safe reinforcement learning via human intervention". In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, pp. 2067–2069.

Schatten, Markus, Jurica Ševa, and Igor Tomičić (2016). "A roadmap for scalable agent organizations in the internet of everything". In: *Journal of Systems and Software* 115, pp. 31–41.

Schrauwen, Benjamin, David Verstraeten, and Jan Van Campenhout (2007). "An overview of reservoir computing: theory, applications and implementations". In: *Proceedings of the 15th european symposium on artificial neural networks. p. 471-482 2007*, pp. 471–482.

Schulman, John et al. (2017). "Proximal policy optimization algorithms". In: *arXiv preprint arXiv:1707.06347*.

Seglen, Per O (1992). "How representative is the journal impact factor?" In: *Research evaluation* 2.3, pp. 143–149.

Seredyński, Franciszek and Jakub Gąsior (2019). "Collective Behavior of Large Teams of Multi-agent Systems". In: *International Conference on Practical Applications of Agents and Multi-Agent Systems*. Springer, pp. 152–163.

Shalev-Shwartz, Shai, Shaked Shammah, and Amnon Shashua (2016). "Safe, multi-agent, reinforcement learning for autonomous driving". In: *arXiv preprint arXiv:1610.03295*.

Shanta, A, AS Pradhan, and SD Sharma (2013). "Impact factor of a scientific journal: is it a measure of quality of research?" In: *Journal of medical physics/Association of Medical Physicists of India* 38.4, p. 155.

Sharifzadeh, Sahand et al. (2016). "Learning to drive using inverse reinforcement learning and deep q-networks". In: *arXiv preprint arXiv:1612.03653*.

Shen, Zhong, Junhan Na, and Zheng Wang (2017). "A biomimetic underwater soft robot inspired by cephalopod mollusc". In: *IEEE Robotics and Automation Letters* 2.4, pp. 2217–2223.

Shepherd, Robert F et al. (2011). "Multigait soft robot". In: *Proceedings of the national academy of sciences* 108.51, pp. 20400–20403.

Shoham, Yoav, Rob Powers, and Trond Grenager (2007). "If multi-agent learning is the answer, what is the question?" In: *Artificial Intelligence* 171.7, pp. 365–377.

Silva, Fernando, Luís Correia, and Anders Lyhne Christensen (2017). "Hyper-learning algorithms for online evolution of robot controllers". In: *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 12.3, p. 14.

Silva, Fernando et al. (2015). "odNEAT: An algorithm for decentralised online evolution of robotic controllers". In: *Evolutionary Computation* 23.3, pp. 421–449.

Silver, David et al. (2014). "Deterministic policy gradient algorithms". In: *ICML*, pp. 387–395.

Silver, David et al. (2016). "Mastering the game of Go with deep neural networks and tree search". In: *Nature* 529, pp. 484–503. URL: http://www.nature.com/nature/journal/v529/n7587/full/nature16961.html.

Sims, Karl (1994). "Evolving virtual creatures". In: *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. ACM, pp. 15–22.

Singh, Amanpreet, Tushar Jain, and Sainbayar Sukhbaatar (2018). "Learning when to communicate at scale in multiagent cooperative and competitive tasks". In: *arXiv preprint arXiv:1812.09755*.

Skrickij, Viktor, Eldar Sabanovic, and Vidas Zuraulis (2020). "Autonomous Road Vehicles: Recent Issues and Expectations". In: *IET Intelligent Transport Systems*.

Stanley, Kenneth O (2007). "Compositional pattern producing networks: A novel abstraction of development". In: *Genetic programming and evolvable machines* 8.2, pp. 131–162.

Stanley, Kenneth O and Risto Miikkulainen (2002). "Evolving neural networks through augmenting topologies". In: *Evolutionary computation* 10.2, pp. 99–127.

Steels, Luc (2000a). "Language as a complex adaptive system". In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 17–26.

— (2000b). "The emergence of grammar in communicating autonomous robotic agents". In.

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement Learning: An Introduction*. A Bradford Book.

Syam, Niladri and Arun Sharma (2018). "Waiting for a sales renaissance in the fourth industrial revolution: Machine learning and artificial intelligence in sales research and practice". In: *Industrial Marketing Management* 69, pp. 135–146.

Szabó, Zoltán Gendler (2017). "Compositionality". In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Summer 2017. Metaphysics Research Lab, Stanford University.

Talamini, Jacopo, Eric Medvet, and Alberto Bartoli (2019). "Communication-based cooperative tasks: how the language expressiveness affects reinforcement learning". In: *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*. ACM, pp. 898–905.

Talamini, Jacopo et al. (2018). "Selfish vs. global behavior promotion in car controller evolution". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*. ACM, pp. 1722–1727.

Talamini, Jacopo et al. (2019). "Evolutionary Synthesis of Sensing Controllers for Voxel-based Soft Robots". In: *Artificial Life Conference Proceedings*. MIT Press, pp. 574–581.

Tan, Ming (1993). "Multi-agent reinforcement learning: Independent vs. cooperative agents". In: *Proceedings of the tenth international conference on machine learning*, pp. 330–337.

Tanaka, Gouhei et al. (2019). "Recent advances in physical reservoir computing: A review". In: *Neural Networks* 115, pp. 100–123.

Todeschini, Roberto and Alberto Baccini (2016). *Handbook of bibliometric indicators: quantitative tools for studying and evaluating research*. John Wiley & Sons.

Tram, Tommy et al. (2018). "Learning negotiating behavior between cars in intersections using deep q-learning". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 3169–3174.

Trimmer, Barry (2013). "Soft robots". In: *Current Biology* 23.15, R639–R641.

Tumova, Jana et al. (2013). "Least-violating control strategy synthesis with safety rules". In: *Proceedings of the 16th international conference on Hybrid systems: computation and control*, pp. 1–10.

Turner, Ralph H, Lewis M Killian, et al. (1957). *Collective behavior*. Vol. 3. Prentice-Hall Englewood Cliffs, NJ.

Van Noorden, Richard (2014). *Publishers withdraw more than 120 gibberish papers, Nature, 24 February 2014*.

Van Raan, Anthony FJ (2006). "Comparison of the Hirsch-index with standard bibliometric indicators and with peer judgment for 147 chemistry research groups". In: *scientometrics* 67.3, pp. 491–502.

Vanholme, Benoit et al. (2013). "Highly automated driving on highways based on legal safety". In: *IEEE Transactions on Intelligent Transportation Systems* 14.1, pp. 333–347.

Vaughan, Neil (2018). "Evolution of Neural Networks for Physically Simulated Evolved Virtual Quadruped Creatures". In: *Conference on Biomimetic and Biohybrid Systems*. Springer, pp. 507–516.

Vrieze, Colin de et al. (2018). "Cooperative Multi-Agent Reinforcement Learning for Low-Level Wireless Communication". In: *arXiv preprint arXiv:1801.04541*.

Wang, Chen, Lin Liu, and Chengcheng Xu (2020). "Developing a New Spatial Unit for Macroscopic Safety Evaluation Based on Traffic Density Homogeneity". In: *Journal of Advanced Transportation* 2020.

Wang, Jane X et al. (2019). "Evolving intrinsic motivations for altruistic behavior". In: *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, pp. 683–692.

Watkins, Christopher JCH and Peter Dayan (1992). "Q-learning". In: *Machine learning* 8.3-4, pp. 279–292.

Wildgaard, Lorna, Jesper W Schneider, and Birger Larsen (2014). "A review of the characteristics of 108 author-level bibliometric indicators". In: *Scientometrics* 101.1, pp. 125–158.

Wilhite, Allen W and Eric A Fong (2012). "Coercive citation in academic publishing". In: *Science* 335.6068, pp. 542–543.

Wu, Cathy et al. (2017). "Flow: A Modular Learning Framework for Autonomy in Traffic". In: *arXiv preprint arXiv:1710.05465*.

Yamada, Tatsuro et al. (2017). "Representation Learning of Logic Words by an RNN: From Word Sequences to Robot Actions". In: *Frontiers in neurorobotics* 11, p. 70.

Yu, Han et al. (2013). "A survey of multi-agent trust management systems". In: *IEEE Access* 1, pp. 35–50.

Zeng, An et al. (2017). "The science of science: From the perspective of complex systems". In: *Physics Reports* 714, pp. 1–73.

Zhang, Chongjie and Victor Lesser (2013). "Coordinating multi-agent reinforcement learning with limited communication". In: *Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, pp. 1101–1108.

Zhang, Chongjie, Victor Lesser, and Sherief Abdallah (2010). "Self-organization for coordinating decentralized reinforcement learning". In: *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, pp. 739–746.

Zhang, Si-Ping et al. (2019). "Collective behavior of artificial intelligence population: transition from optimization to game". In: *Nonlinear Dynamics* 95.2, pp. 1627–1637.