Evolutionary Robotics and Robotic Systems

Literature Review

Ryan Scott
University of Cape Town
Cape Town, South Africa
rqscott97@gmail.com

ABSTRACT

Evolutionary robotics has been a useful tool now for some time in allowing natural evolution principles and mechanisms to be applied to simulated robotic environments. The techniques and methods expand upon established neural network bases and apply the tools of machine learning to ‘evolve’ the robots in terms of morphology and/or controls while applying mutations and/or crossover functions to adapt and transform the robots in an evolutionary way. This can allow much faster optimization or solution finding than manually testing different configurations or parameters. This review serves to provide a summary of recent history of evolutionary robotics, as well as provide brief descriptions of recent techniques and methodologies used in the field.

KEYWORDS

Evolutionary Robotics, Evolutionary Algorithms, Evolutionary Computation, Machine Learning, Neural Networks

1 Introduction

The field of Evolutionary Robotics attempts to utilize simulation and machine learning techniques in tandem with artificial life and biological evolutionary principles of natural selection to find optimal parameters satisfying a fitness function of some kind [1,4,5,6]. This has been proven to be a useful tool to generate optimal robots and robot controllers for at least certain classes of robots [2]. Even in situations where ER (Evolutionary Robotics) does not show results that map to real life effectively, we gain valuable knowledge and insight through both the process and the results of using the technique [3]. It can also show proof of concept and allows exploratory studies quite effectively [3].

2 Neuro-Evolution

This section provides a brief overview of the techniques, tools and algorithms used in the field of Neuro-Evolution. It finishes up with some seminal works as well as recent findings in the field. Neuro-evolution can be summarized as the application of evolutionary algorithms to train artificial neural networks [28]. The type of algorithms used is an Evolutionary Algorithm (as described below in 2.1.2). The primary idea of neuro-evolution is to model the process whereby a ‘brain’ learns or ‘evolves’ at becoming better at a task. This technique has become widely applicable

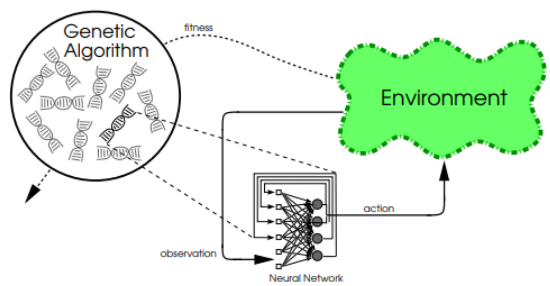

Fig 1. Neuro-evolution. [48] Individuals are transformed before being entered into the neural network, outputting an action which is then evaluated in the environment with a fitness function.

being used in areas ranging from traffic congestion [29] to the AI in simulations and videogames [30,31].

2.1 Artificial Neural Networks (ANN)

A field that can be seen to originate from as early as the 1940s [12], the idea of an Artificial Neural Network has been around for some time. As a construct, it can be said to be a network of connected neurons that each take input, process, and output. These networks are usually trained in such a way that each neuron (or node)’s edge weightings are adjusted at each iteration. Typically, this has been done with what is called Backpropagation, but alternatives inspired by evolutionary processes have been proposed as alternatives for training Neural Networks[13][14]. The networks are organized in a system of layers of neurons, consisting of one input layer, one output layer, and anywhere from 0 to n interior layer. Each layer’s nodes also have weights assigned to it between edges of the nodes of this layer and adjacent layers. At each layer, values are received from the previous layer and directed into the current layers’ neurons based on edge weightings. Within these neurons, a weighted sum of inputs is calculated typically called the Transfer Function. This value is then passed to an Activation Function (of which several exist, most commonly used past being the Sigmoid function but with experimentation with different functions being done as far back as more than 1990 if not further[15]). The Activation

Function will then determine which nodes to fire in the following layer. Eventually, the output layer will be reached, which should yield some final answer. ANN's are typically the backbone of any learning method, and have been used often in the field of Robotics in general [18,19]

2.1.1 Activation Functions. Many activation functions exist, some popular ones are shown below. The step function is binary in that it simply fires if a certain threshold is reached. If the given Transfer Function presents a certain output > some x, then the neuron fires. Else, do not fire. The Sigmoid function is non-linear, smooth and can be easily related to some other functions [17] making it a very useful tool for certain problems. In addition, it can also introduce nonlinearity to the model, and it is computationally easy to perform as well as it's derivative. As such, derivatives of the sigmoid function are often employed in learning algorithms.
Some other common activation functions exist, such as the tanh and ReLu functions, each having their use.

2.1.2 Training Methods. Just as in activation functions, many different training methods exist. Gradient Descent[20] is one of the most common archetypes used. Briefly summarized, this technique uses Backpropagation to track back along edge weightings and adjust them based on a computed error. In this way, the network slowly 'descends' towards a desired solution. There is some challenges within Gradient Descent and Backpropagation methods [20,21,22,24] mostly due to it's nature of settling around points that are suboptimal local minima, yet deceptively close to the solution. It is argued that these are saddle points usually surrounded by a plateau of same error, making it notoriously difficult for Gradient Descent to escape [21]. Recent work has shown that Gradient Descent in large enough dimension with large number of global minima will always end in a fake minimum located close to the global minimum [24]. Nevertheless, Gradient Descent is still used often even today [23,25]. Simulated Annealing[26] is an alternative to Gradient Descent methods that handles the local minima problem very aptly. The method works by 'heating' the model and slowly lowering the 'temperature' to minimize system energy. Many variations of this method exist, just as in Gradient Descent.
A third popular training method is the class of Evolutionary Algorithms[27]. These algorithms will typically simulate elements of natural evolutionary processes in order to 'evolve' the network and it's weightings. Some work has established that EA (Evolutionary Algorithms) produce solutions equal to or better than simulated annealing. In addition, while both Gradient Descent and simulated annealing are typically only able to find single solutions (global minimum), Evolutionary Algorithms can be used to find a set of Pareto-Optimal solutions, should they exist. Evolutionary Algorithms are thus far better than Gradient Descent and Simulated Annealing for these class of problems.

2.2 Multi-Objective Neuro-Evolution

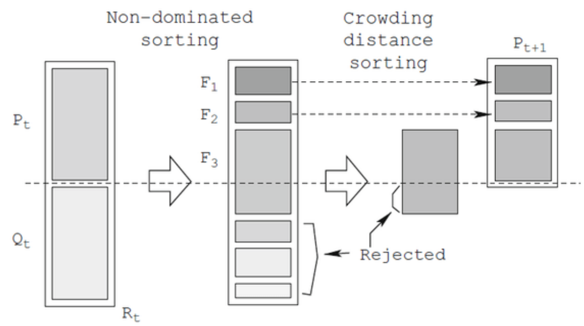

Fig 2. NSGA-II process[37]

Multi-Objective Neuro-Evolution refers to techniques used to find a set of non-dominated solutions where these solutions will be based on conflicting objectives, with trade-offs.[32,34]

2.2.1 NSGA-II. NSGA-II (Non-dominated Sorting Genetic Algorithm II) is a fast and elitist multiobjective genetic algorithm designed to alleviate 3 typical criticisms of Multi-objective evolutionary algorithms (MOEAs)[33]. **Fig 2.** demonstrates quite nicely how it iterates. The criticisms are:

- Terrible complexity at O(MN^3) (where M is number of objectives and N pop size).
- Non-elitism approach (elitism refers to being able to copy an already incredibly fit subset of population into the next iteration of the genetic algorithm. This prevents the algorithm from wasting time rediscovering. It is shown that elitism can speed up performance [34,35].
- The need of a sharing parameter (a sharing parameter will specify how far away 2 individuals must be so to decrease each other's fitness).

NSGA-II performs according to the following steps:

1. Sort the combination of parent and offspring (if existing) populations in order of ascending level of non-domination.
2. Organize the results into several discrete rankings, classified by fronts.
3. Fill a new population according to front ranking.
4. If one front is taken partially, perform Crowding-sort which sorts based on crowding distance related with the density of solutions. Prioritize less dense.
5. Create new offspring population from this population using crowded tournament selection (which compares first by front ranking, then by crowding distance), crossover and mutation operators.

NSGA-II has notably better performance than most other methods in its class [33]. Chen[42] improved the NSGA-II to allow its non-dominance concept to extend to the constraint space. Gong et al. created a fast optimized algorithm based on NSGA-II that was inspired by Mode-Matching Technique (MMT)[43].

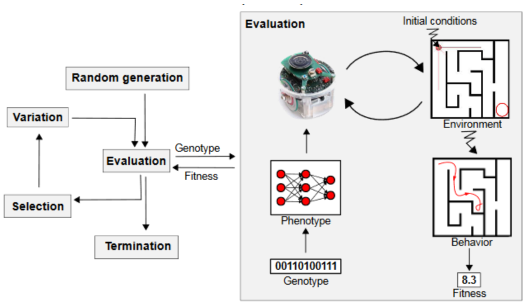

Fig 3. Principle operation of Evolutionary Robotics[7]

2.2.2 Evolution Strategies. Evolution Strategies (ES) differs from Genetic Algorithms(GA) in that it utilizes mutation as it's main reproduction factor, if not it's only one. Modern ES variants are based on the capability to self-adapt the internal strategy parameters. An example of this is multi-objective covariance matrix adaptation evolution strategy (MO-CMA-ES)[39]. In Evolution Strategy, new individuals are sampled according to a multivariate normal distribution. Recombination is simply selecting a new mean value for the distribution, and mutation is adding a random vector, that is a perturbation with a mean of zero. Pairwise dependencies between variables are represented by a covariance matrix, and the covariance matrix adaptation is a method which is used to update the covariance matrix of this distribution. Compared to NSGA-II for certain problems, MO-CMA-ES is better at finding larger sub-sets of approximated Pareto-optimal solutions, whereas NSGA-II is better a complimentary sub-set of optimal solutions.[41]

3. Evolutionary Robotics

This section brings together the previous sections' features into the field of Evolutionary Robotics, and displays some of it's features and common techniques, as well as seminal and recent works done in the field.

3.1 Main Principles

The main principles of Evolutionary Robotics can be summarized quite concisely in the diagram **Fig 3.** These overlap with evolutionary ideas of 'survival of the fittest', and this is by design, as ER is inspired by biological ideas of evolution. In each iteration, a genotype is generated and tested with a fitness function, as shown on the right side of the diagram. This repeats until some exit condition, or manually, whichever termination condition is designed. These principles have been fairly established now for some time [10]. The problems that ER mostly concerns itself with are that which are discerned from the nature of autonomous robots – the connections between parameters and variables are complex and not well defined, often containing much noise. In addition, the solution space will often have conflicting constraints and objectives. Evolutionary algorithms are well suited for this class of problems then, as they don't require perfectly defined fitness functions, can handle multiple constraints and objectives, and can find adequate solutions even under noisy and dynamic conditions [11]. The field of evolutionary robotics can be said to be founded in the early 1990s with seminal work [45,46,47]. Cliff, Husbands and Harvey strongly advocated for Artificial Evolution to be simulated in the design of robotic controls, and provided strong evidence to support this [45]. Beer and Gallagher demonstrated evolving artificial neural networks with a GA in order to obtain real results with walking robots [46]. Nolfi et al. were some of the first to outline some distinctly different approaches to developing neural controllers using evolutionary robotics techniques[44]. They showed that interestingly enough, it is feasible to evolve a robot in a physical environment. They admit that this it time-consuming though, and that simulated approaches are thus still useful.

This review will narrow itself to focusing mainly on the area of Evolutionary Collective Robotics.

3.2 Evolutionary Collective Robotics

In this section an overview of the area of Evolutionary Collective Robotics [49] will be given, from some early seminal work to what is being done recently and today.

The field can be seen to have begun when Ficici et al. [50] coined the phrase *embodied evolution* and discussed evolutionary processes that are distributed over robots in a population to allow them some autonomous and continuous adaptation to environments. The robots are autonomous at 2 levels: autonomy in terms of performing tasks without external control or oversight and autonomy in terms of improvement and adaptation (through evolutionary processes) to its environment. This helps us in problems where the complexity of interactions between robots or complexity of environment makes the scenario predictions too inaccurate or with too wide a margin of accuracy.

A reality gap is the idea that simulation techniques do not model reality accurately enough to be useful, and embodied evolution or onboard adaptivity avoids this problem [51]. Also, embodied evolution or evolutionary collective robotics is naturally parallelizing the evolutionary process, which can provide benefits, even such as superlinear speedups [52]. In terms of pure task performance, collective robotics techniques have been shown to outperform alternative evolutionary robotics techniques in areas such as surveillance and self-localization [53,54].

Bredeche et al. [49] summarized the unique features of collective robotics quite concisely:

Decentralized. There is no central controller or authority asserts the overall evolutionary procedure. Robots will assess and perform evolutionary iteration using their own local information and abilities.

Online. Evolution occurs during the operation of the robot, while it is 'online' and in it's task environment. The process continues after the robot is deployed.

Parallel. Due to the nature of having distributed evolutionary platforms on multiple online robots, the robots will perform and evolve in the same environment concurrently with one another. They will interact as well, allowing them to exchange genetic information locally.

	Implementation				Robot behavior					Experimental settings		Mating conditions			Selection scheme			Replacement scheme			
	Distributed	Hybrid	Simulation	Real robots	Monomorphic	Polymorphic	Individual behavior	Cooperation	Division of labor	Task	No of robots	Panmictic	Proximity	Other	Performance	Random	Genotypic distance	Fixed lifetime	Variable lifetime	Event based	Limited lifetime
Ficici et al. (1999); Watson et al. (2002)	•			•	•		•			Phototaxis	8		•		•				•		
Simões and Dimond (2001)	•			•	•		•			Obstacle avoidance	6	•			•			•			
Usui and Arita (2003)		•		•	•		•			obstacle avoidance	6		•		•			•			
Bianco and Nolfi (2004)	•		•		•		•			Self-assembly	64		•			•				•	
Hettiarachchi et al. (2006); Hettiarachchi and Spears (2009)	•		•		•		•			Navigation with obstacle avoidance	60		•		•					•	
Wischmann et al. (2007)	•		•		•		•			Foraging[a]	3		•		•				•		
Perez et al. (2008)		•	•		•		•			Obstacle avoidance	5	•			•			•			
König and Schmeck (2009); König et al. (2009)	•		•		•		•			Obstacle avoidance with gate passing	26; 30		•		•			•			
Pugh and Martinoli (2009)		•	•	•	•		•			Obstacle avoidance	1–10	•	•		•			•			
Prieto et al. (2009); Trueba et al. (2011, 2012)	•		•			•		•	•	Surveillance, foraging, construction	20		•		•					•	•
Bredeche and Montanier (2010, 2012) Bredeche (2014)	•		•	•	•			•		None	20; 4000		•			•					•
Prieto et al. (2010)	•			•		•		•	•	Surveillance	8		•		•					•	•
Schwarzer et al. (2010)	•		•	•		•		•		None	Up to 40		•			•			•		
Schwarzer et al. (2011)	•		•		•		•			Phototaxis	4		•		•			•			
Montanier and Bredeche (2011, 2013)	•		•		•			•		None	100		•			•	•				•
Huijsman et al. (2011)	•	•	•		•		•			Obstacle avoidance	4–400	•		•	•			•			
Karafotias et al. (2011)		•	•		•		•		•	Obstacle avoidance, phototaxis, and patrolling	10			•	•			•			
Silva et al. (2012, 2013, 2015, 2017)		•	•	•		•	•	•		Navigation, aggregation, surveillance, and phototaxis	2–20		•		•					•	
Weel et al. (2012a,b)		•	•		•			•		Foraging	10; 50		•		•			•			
García-Sánchez et al. (2012)		•	•		•		•			Obstacle avoidance	4–36		•	•	•			•			
Haasdijk and Bredeche (2013) Haasdijk et al. (2013, 2014a); Noskov et al. (2013); Haasdijk and Eigenhuis (2016); Bangel and Haasdijk (2017); Kemeling and Haasdijk (2017)	•		•			•	•		•	Foraging	100		•		•			•			
Trueba et al. (2013) Trueba (2017)	•		•	•		•		•	•	Synthetic mapping, gathering, self-location	40; 20; 9	•	•		•	•				•	•
O'Dowd et al. (2014)		•		•	•		•			Foraging	10		•		•			•			
Fernandez Pérez et al. (2014)	•		•		•				•	Foraging	50		•		•	•		•			
Fernandez Pérez et al. (2015)	•		•		•			•		Foraging	100		•		•					•	
Hart et al. (2015); Steyven et al. (2016)	•		•		•			•		Foraging	100		•		•						•
Heinerman et al. (2015, 2016)		•	•	•	•		•			Obstacle avoidance	6	•			•			•			
Montanier et al. (2016); Bredeche et al. (2017)	•		•			•			•	Foraging	100; 500		•		•	•					•
Fernandez Pérez et al. (2017)	•		•			•		•		Foraging	200		•		•			•			

[a]*As a proxy for predator avoidance.*

Fig 4. Collective & Embodied Robotics Research [49]

Because of the ability of robots to evolve through exchanging of information, this allows the robots to evolve in a new way, through *mating*.

Mating. Robots (2 or more) exchange genetic material, which might be used to produce offspring.

3.2.1 Algorithmic Process. The algorithmic process for the evolution of collective robots can be generalized into the following steps.

1. *Initialization*. The robot controllers are initialized randomly according to some parameters.
2. *Function*. The robots will peruse and act within the environment around them as per their robotic behavior predefined. This may happen in parallel with the other following steps.
3. *Calculate Fitness*. The robots will self-evaluate their performance based on predefined notions of fitness and what constitutes good metrics.
4. *Compare Fitness*. The genomes are then evaluated and compared for fitness.
5. *Mating*. The essential step in collective robotics, robots will exchange genetic material to create a pool of candidate parents from the robots to be considered in the parent selection process.
6. *Replacement*. The active genome is replaced by a new individual. This could be triggered by internal conditions, or interactions [55].
7. *Parent selection*. Genetic information is chosen for the creation of new offspring from received genetic information through mating events. If there is an objective, usually the performance on this objective is used for the basis of selection.
8. *Variation*. A new genome is created using variation operators on selected parents.

It is reassuring to side-note that the notion that simulation is an effective tool for transferring its results into real robots, and performs as good if not better than real physical experiments, is validated even recently today [59].

3.2.2 State of the Art and Cost. Here we discuss recently published works. Physical implementations will not be looked into too deeply as they extend beyond the scope of this paper. The table in **Fig.4** neatly summarizes contributions to the field, which may cover several papers. These are all relating to embodied collective robotics since the seminal work of Ficci et al. in 1999. There are some other recent works worth mentioning not on the table. Baldassare et al. [57] showed that utilizing self-organizing behavior and interactions between robots is a strong technique to simulate collective behavior. Haasdijk [56] developed MONEE (Multi-Objective aNd open-Ended Evolution), a platform that then let him explore trade-offs between environment- and task-driven selection pressures. He found that task-based selection exerts higher pressure than the environment quantitatively. The field is clearly becoming more mature and established, with the creation of dedicated reusable software [58] and frameworks [56]. Nitscke, G[60] experimentally compared two popular Neuro-Evolution techniques, NEAT (Neuro-Evolution for Augmenting Topologies) and CONE-2 (Collective Neuro-Evolution 2) in a collective construction task. It was found that CONE-2 teams achieved a higher average team fitness compared to that of NEAT, in environments necessitating cooperation of two robots or more. It showed that CONE-2 is a better method for encouraging behavioral specialization, which would then encourage cooperation in a collective or team of robotics. Watson and Nitshke[61] then went further and then did a broad study of methods to adapt behaviors and morphologies of simulated collective robot teams. Throughout it, HyperNEAT [62], an adaptation of NEAT. HyperNEAT is special due to it's encoding, called Compositional Pattern Producing Networks (CPPN), which in practice lends itself to efficiently evolving very large (i.e. larger than other methods) neural networks which more accurately represent the brain. HyperNEAT also has the advantage of 'seeing' or modelling the problem domain, which strongly favors itself for robotics. Most recently, Furman et al.[63] investigated for the first time a cost (fitness cost) of complexity, evaluating the relationship between task difficulty (environment complexity) and morphological complexity. They found evidence that effective behaviors in increasingly complex environments does not necessarily require increased morphological complexity. They acknowledge that although this contradicted previous research [64], this may be due to differing definitions. The paper finishes by stating that this specific new area of complexity costs and relationships related to it is an open area of ongoing research.

4. CONCLUSIONS

The field of Evolutionary Robotics overall has matured greatly since it's beginnings in the 1990s. The power of evolutionary methods and their ability to be applied to real world robots is well established now.

More narrowly, the subfield of Evolutionary Collective Robotics has matured recently as well, with frameworks and software suites being available specifically for them. This indicates a trend of growing interest in the field which may continue.

There are however significant gaps in the research however, surrounding complexity costs within robotics and relationships between morphological complexity and effective behaviors and performance. Specifically, the lack of investigation into the effects of imposing a constraint onto complexity (morphological or control), to simulate the energy requirements of evolution. This necessitates future research into this previously intuitively answered question, which state-of-the-art findings have begun to dispute.

An investigation using perhaps multiple different multi-objective neuro-evolution methods (such as HyperNEAT) to test the effects of imposing energy cost constraints on individuals would serve this field well.

REFERENCES

[1] Stefano Nolfi, Dario Floreano and Director Dario Floreano. 2000. *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines* (1^{st}. ed.). MIT press.

[2] Nick Jacobi, Phil Husbands and Inman Harvey. 1995. Noise and the reality gap: The use of simulation in evolutionary robotics. In *Proceedings of the Third European Conference in Artificial Life*, 704-720.

[3] Inman Harvey, Ezequiel Di Paolo, Rachel Wood, Matt Quinn and Ello Tuci. 2005. Evolutionary Robotics: A New Scientific Tool for Studying Cognition. *Artificial Life*, 11(1-2), 79–98.

[4] Stephane Doncieux, Nicolas Bredeche, Jean-Baptiste Mouret and Agoston E. Gusz Eiben. 2015. Evolutionary robotics: what, why, and where to. In *Frontiers in Robotics and AI 2*, 4.

[5] Takashi Gomi and Ann Griffith. 1996. "Evolutionary robotics-an overview," In *Proceedings of IEEE International Conference on Evolutionary Computation* Nagoya, Japan, 40-49.

[6] Stephane Doncieux, Jean-Baptiste Mouret, Nicholas Bredeche and Vincent Padois. 2011. Evolutionary Robotics: Exploring New Horizons. In *New Horizons in evolutionary robotics*. Springer, Berlin, Heidelberg, 3–25.

[7] Nicholas Bredeche, Stephane Doncieux and Jean-Baptiste Mouret. 2018. Evolutionary robotics tutorial. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion on - GECCO* '18.

[8] Richard A. Watson, Sevan G. Ficici and Jordan B. Pollack. 2002. Embodied Evolution: Distributing an evolutionary algorithm in a population of robots. In *Robotics and Autonomous Systems 39*, 1 (April 2002), 1-18. DOI: https://doi.org/10.1016/S0921-8890(02)00170-7

[9] Karl Sims. 1994. Evolving 3D morphology and behavior by competition. *Artificial life 1*, 4, 353-372.

[10] Thomas Back and H-P. Schwefel. 1996. Evolutionary computation: an overview. In *Proceedings of IEEE International Conference on Evolutionary Computation*. IEEE, 20-29. DOI: 10.1109/ICEC.1996.542329

[11] Agostan E. Eiben and James E. Smith. 2015. Introduction to Evolutionary Computing. Natural Computing Series, Vol 53. Berlin, Springer. DOI: 10.1007/978-3-662-44874-8

[12] Warren S. McCulloch and Walter Pitts. 1943. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics* 5, 4, 115-133.

[13] Jarmo Ilonen, Joni-Kristian Kamarainen and Jouni Lampinen. 2003. Differential Evolution Training Algorithm for Feed-Forward Neural Networks. *Neural Processing Letters* 17,1, 93-105. DOI: https://doi.org/10.1023/A:1022995128597

[14] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor and Ilya Sutskever. 2017. Evolution Strategies as a Scalable Alternative to Reinforcement Learning. arXiv:1703.03864. Retrieved from https://arxiv.org/abs/1703.03864.

[15] Donald F. Specht. 1990. Probabilistic neural networks. *Neural Networks* 3, 1, 109–118. DOI:10.1016/0893-6080(90)90049-q

[16] Sharma V Avinash. 2017. Understanding Activation Functions in Neural Networks. Retrieved April 22, 2019 from https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0

[17] Taskin Kocak. 2015. Sigmoid Functions and Their Usage in Artificial Neural Networks. *School of Electrical Engineering and Computer Science, presentation*.

[18] Naoyuki Kubota. 2002. Perception-based robotics based on perceiving-acting cycle with modular neural networks. In *Proceedings of the 2002 International Joint Conference on Neural Networks*. Honolulu, HI, USA, 477-482. DOI: 10.1109/IJCNN.2002.1005519

[19] Neha Joshi, Anil Kumar, Pavan Chakraborty and Rahul Kala. 2015. Speech controlled robotics using Artificial Neural Network. In *Third International Conference on Image Information Processing (ICIIP)*. Waknaghat, 526-530. DOI: 10.1109/ICIIP.2015.7414829

[20] Sebastian Ruder. 2016. An overview of gradient descent optimization algorithms. Retrieved April 23, 2019 from http://ruder.io/optimizing-gradient-descent/

[21] Yann N. Dauphin, Razvan Pascanu, Calgar Gulcehre, Kyunghyun Cho, Surya Ganguli and Yoshua Bengio. 2014. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. arXiv:1406.2572. Retrieved from http://arxiv.org/abs/1406.2572

[22] Carlos E. Perez. 2017. Why we should be Deeply Suspicious of BackPropogation. Medium. Retrieved April 23, 2019 from https://medium.com/intuitionmachine/the-deeply-suspicious-nature-of-backpropagation-9bed5e2b085e

[23] Suhas Lohit, Dehong Liu, Hassan Mansour and Petros T. 2019. Unrolled Projected Gradient Descent for Multi-spectral Image Fusion. In *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton, United Kingdom, 7725-7729. DOI: 10.1109/ICASSP.2019.8683124

[24] Anne Bouillard and Philippe Jacquet. 2019. Quasi Black Hole Effect of Gradient Descent in Large Dimension: Consequence on Neural Network Learning. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Brighton, United Kingdom, 8365-8369. DOI: 10.1109/ICASSP.2019.8683250

[25] Rishabh Dixit, Amrit Singh Bedi, Ruchi Tripathi and Ketan Rajawat. 2019. Online Learning With Inexact Proximal Online Gradient Descent Algorithms. In *IEEE Transactions on Signal Processing* 67, 5, 1338-1352. DOI: 10.1109/TSP.2018.2890368

[26] Dimitris Bertsimas, John Tsitsiklis. 1993. Simulated annealing. In *Statistical science* 8, 1, 10-15. Retrieved April 23, 2019 from http://www.mit.edu/~dbertsim/papers/Optimization/Simulated%20annealing.pdf

[27] Lawrence Davis. 1991. Handbook of genetic algorithms. Von Nostrand Reinhold, New York.

[28] Dannielle Whittaker. 2012. Evolution 101: Neuroevolution. Retrieved April 23, 2019 from https://beacon-center.org/blog/2012/08/13/evolution-101-neuroevolution/

[29] Ming-Kai Jiau and Shih-Chia Hunag. 2018. Self-Organizing Neuroevolution for Solving Carpool Service Problem With Dynamic Capacity to Alternate Matches. In *IEEE Transactions on Neural Networks and Learning Systems* 30, 4, 1048-1060 DOI: 10.1109/TNNLS.2018.2854833

[30] Kenneth O. Stanley, Bobby D. Bryant and Risto Miikkulainen. 2005. Real-time neuroevolution in the NERO video game. In *IEEE Transactions on Evolutionary Computation* 9, 6, 653-668. DOI: 10.1109/TEVC.2005.856210

[31] Luigi Cardamone, Daniele Loiacono and Pier Luca Lanzi. 2010. Learning to Drive in the Open Racing Car Simulator Using Online Neuroevolution. *In IEEE Transactions on Computational Intelligence and AI in Games* 2, 3, 176-190.DOI: 10.1109/TCIAIG.2010.2052102

[32] Kar Bin Tan, Jason Teo and Patricia Anthony. 2011. A multi-objective neuro-evolutionary optimization approach to intelligent game AI synthesis. In *ICIMU 2011 : Proceedings of the 5th international Conference on Information Technology & Multimedia*, 1-5. DOI: 10.1109/ICIMU.2011.6122716

[33] Kalyanmoy, Amrit Pratap, Sameer Agarwal and T.A.M.T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. In *IEEE Transactions on Evolutionary Computation* 6, 2, (April 2002), 182-197. DOI: 10.1109/4235.996017

[34] Carlos M. Fonseca and Peter J. Fleming. 1993. Genetic Algorithms for Multiobjective Optimization: FormulationDiscussion and Generalization. In *Icga* 93, (July 1993), 416-423. DOI: 10.1.1.48.9077

[35] Günter Rudolph. 1999. Evolutionary search under partially ordered sets. Dept. Comput. Sci./LS11, Univ. Dortmund, Dortmund, Germany, Tech. Rep. CI-67/99.

[36] Eckart Zitzler, Kalyanmoy Deb and Lothar Thiele. 2000. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation* 8, 2, 173-195.

[37] Kalyanmoy Deb. 2001. Multi-objective optimization using evolutionary algorithms, Vol. 16. John Wiley & Sons.

[38] Ruben Putter and Geoff Nitschke. 2017. Evolving morphological robustness for collective robotics. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)* Honolulu, HI, 1-8. DOI: 10.1109/SSCI.2017.8285267

[39] Christian Igel, Mikolaus Hansen and Stefan Roth. 2007. Covariance matrix adaptation for multi-objective optimization. In *Evolutionary computation 15*, 1, 1-28.

[40] Thomas Voß, Nicola Beume, Gunter Rudolph and Christian Igel. 2008. Scalarization versus indicator-based selection in multi-objective CMA evolution strategies. In *IEEE Congress on Evo-lutionary Computation 2008 (CEC 2008)*. IEEE Press (2008), 3041–3048.

[41] Amiram Moshaiov and Omer Abramovich. 2014. Is MO-CMA-ES superior to NSGA-II for the evolution of multi-objective neuro-controllers? In *2014 IEEE Congress on Evolutionary Computation (CEC)*. Beijing, 2809-2816. DOI: 10.1109/CEC.2014.6900433

[42] Jianling Chen. 2009. Multi-Objective Optimization of Cutting Parameters with Improved NSGA-II. In *2009 International Conference on Management and Service Science*. Wuhan, 1-4. DOI: 10.1109/ICMSS.2009.5302835

[43] Jinjie Gong, Xiaofang Zhu, Quan Hu, Yulu Hu and Bin Li. 2017. A fast optimized algorithm based on the NSGA — II for microwave windows. In *2017 Eighteenth International Vacuum Electronics Conference (IVEC)*. London, 1-2. DOI: 10.1109/IVEC.2017.8289508

[44] Stefano Nolfi, Dario Floreano, Orazio Miglino and Francesco Mondada. 1994. How to evolve autonomous robots: Different approaches in evolutionary robotics. In *Artificial life iv: Proceedings of the fourth international workshop on the synthesis and simulation of living systems*. MIT Press, 190-197. DOI: https://doi.org/10.3929/ethz-a-010111550

[45] Dave Cliff, Phill Husbands and Inman Harvey. 1993. Explorations in evolutionary robotics. In *Adaptive behavior 2*, 1, 73-110. DOI: https://doi.org/10.1177/105971239300200104

[46] Randall D. Beer and John C. Gallagher. 1992. Evolving Dynamical Neural Networks for Adaptive Behavior. In *Adaptive Behavior 1*, 1, 91–122. DOI: https://doi.org/10.1177/105971239200100105

[47 Rodney A. Brooks. 1992. Artificial life and real robots. *In Proceedings of the First European Conference on artificial life.* 3-10. Retrieved April 25, 2019 from

https://books.google.co.za/books?hl=en&lr=&id=pWsNJkdZ4tgC&oi=fnd&pg=PA3&dq=related:xSKS5ofiuq8J:scholar.google.com/&ots=86I3gZrORB&sig=DTS471__a7HVbSPOnUASaWobpTI
[48] Faustino John Gomez and Risto Miikkulainen. Robust Non-Linear Control through Neuroevolution. Ph.D. Dissertation. The University of Texas, Austin.
[49] Nicolas Bredeche, Evert Haasdijk and Abraham Prieto. 2018. Embodied Evolution in Collective Robotics: A Review. *Front. Robot. AI, 22 February 2018.* Doi: https://doi.org/10.3389/frobt.2018.00012
[50] Seven G. Ficici, Richard A. Watson, and Jordan B. Pollack. 1999. "Embodied evolution: A response to challenges in evolutionary robotics." *Proceedings of the eighth European workshop on learning robots*. 14-22.
[51] Nick Jakobi, Phil Husbands, and Inman Harvey. 1995. "Noise and the reality gap: The use of simulation in evolutionary robotics." In *European Conference on Artificial Life*. 704-720.
[52] Enrique Alba. 2002. "Parallel evolutionary algorithms can achieve super-linear performance." *Information Processing Letters* 82.1. 7-13.
[53] Marijn C. Schut, Evert Haasdjik and Abraham Preito. 2009. Is situated evolution an alternative for classical evolution? In *IEEE Congress on Evolutionary Computation, CEC 2009*, 2971–2976.
[54] Abraham Prieto, Francisco Bellas, Pedro Trueba and Richard J. Duro. 2016. Real-time optimization of dynamic problems through distributed embodied evolution. In *Integr. Comput. Aided Eng. 23*. 237–253. DOI:10.3233/ICA-160522
[55] Richard A. Watson, Sevan G. Ficici and Jordan B. Pollack. 2002. Embodied evolution: distributing an evolutionary algorithm in a population of robots. In *Rob. Auton. Syst. 39*. 1–18. DOI:10.1016/S0921-8890(02)00170-7
[56] Evert Haasdijk. 2015. Combining Conflicting Environmental and Task Requirements in Evolutionary Robotics. In *2015 IEEE 9th International Conference on Self-Adaptive and Self-Organizing Systems*. Cambridge, MA, 131-137. DOI: 10.1109/SASO.2015.21
[57] Gianluca Baldassarre, Stefano Nolfi, and Domenico Parisi. 2003. Evolving mobile robots able to display collective behaviors. In *Artificial life 9*, 3, 255-267.
[58] Nicolas Bredeche, Jean-Marc Montanier, Berend Weel, and Evert Haasdijk. 2013. Roborobo! a fast robot simulator for swarm and collective robotics. arXiv:1304.2888. Retrieved from https://arxiv.org/abs/1304.2888
[59] Miguel Duarte, Vasco Costa, Jorge Gomes, Tiago Rodrigues, Fernando Silva, Sancho Moura Oliveira, and Anders Lyhne Christensen. 2016. Evolution of collective behaviors for a real swarm of aquatic surface robots. In *PloS one 11*. 3.
[60] Geoff Nitschke. 2012. Behavioral heterogeneity, cooperation, and collective construction. In *2012 IEEE Congress on Evolutionary Computation*. Brisbane, QLD, 1-8. DOI: 10.1109/CEC.2012.6256158
[61] James Watson and Geoff Nitschke. 2015. Evolving Robust Robot Team Morphologies for Collective Construction. *2015 IEEE Symposium Series on Computational Intelligence*. Cape Town, 1039-1046. DOI: 10.1109/SSCI.2015.150
[62] Kenneth O. Stanley, David B. D'Ambrosio, and Jason Gauci. 2009. A hypercube-based encoding for evolving large-scale neural networks. In *Artificial life 15*. 2, 185-212.
[63] Alexander Furman, Danielle Nagar, Geoff Nitschke. 2019. The Cost of Complexity in Robot Bodies.
[64] Joshua E. Auerbach and Josh C. Bongard. 2014. Environmental influence on the evolution of morphological complexity in machines. In *PLoS computational biology*, vol. 10, 1.