Georgios Methenitis

Master Thesis

Evolution of Soft Robots by Novelty Search

November 2014

University Of Amsterdam

Master Artificial Intelligence

Master Thesis

Evolution of Soft Robots by Novelty Search

Author:
Georgios Methenitis

Supervisors:
Arnoud Visser, UvA
Dario Izzo, ESA-ESTEC
Daniel Hennes, ESA-ESTEC

ECTS: 42

Submitted to the Board of Examiners in partial fulfillment of the requirements for the degree of
MSc. Artificial Intelligence
of the
University of Amsterdam.

November 2014

"Evolve solutions; when you find a good one, don't stop."

David Eagleman, *Incognito: The Secret Lives of the Brain*

UNIVERSITY OF AMSTERDAM
Faculty of Science

Master Artificial Intelligence

Abstract

Evolution of Soft Robots by Novelty Search

by Georgios Methenitis

Soft robotics is a vivid research field on the science and engineering aspects of soft materials in mobile machines. Recent development in soft robotics and evolutionary optimization have shown the ability to simultaneously evolve the morphology and locomotion of soft robots. Generative encoding coupled with neural evolution of augmented topologies shows promising results. Novelty search, unlike traditional optimization methods does not aim to optimize the objective but instead looks for novelty. Novelty search rewards diversity and leads to a variety of solutions, mimicking natural evolution. Apart from the performance comparison between novelty and fitness based search, this thesis shows that new locomotion patterns can be produced by the former while different types of selection algorithms for fitness and novelty based evolution are studied. In addition, a method to combine both is proposed. Finally, the objective-wise performance is tested under variant gravity conditions leading into a taxonomy of possible locomotion strategies given different gravity levels.

Acknowledgements

The present work has been conducted in association with the *Advanced Concepts Team* (ACT) in *European Space Agency* (ESA-ESTEC, Noordwijk). The subject of this thesis had been offered as an internship project by the ACT under the title: "*Simultaneous evolution of morphology and locomotion of soft robots by novelty search*". Most of this work has been conducted during my 3-month internship at the ACT, ESA-ESTEC.

It would have been impossible to write this thesis without the help and support of my three supervisors, Arnoud Visser (Senior Lecturer, UvA), Dario Izzo (Scientific Coordinator, ACT) and Daniel Hennes (Postdoctoral Research Fellow, ACT).

I would like to thank Arnoud for his valuable comments in the whole duration of this thesis. I am grateful for all the discussion we had the last two years regarding my thesis, Dutch Nao Team, and the number of projects and papers we have done together.

I am most grateful to Dario and Daniel. Their ideas and discussions we had were valuable for me to finish this work. It has been an honor working under their supervision, as I gained too much from them.

I express my warm thanks to all the members of Dutch Nao Team, the Robocup SPL team of University of Amsterdam, in which I was proud to serve as a member for two years. I also thank all the members of ACT who welcomed me in the team and made my three-month internship a wonderful experience. Special thanks to Paul for our "evolutionary" conversations.

I would like to express my gratitude towards everyone who supported me during my master studies. Especially, my family, my friends, and my girlfriend. You have given me your unequivocal support throughout this work and my studies.

Contents

List of Figures

List of Tables

List of Algorithms

This work is dedicated to my family. Thank you for your support during all the years I have been a student.

Chapter 1

Introduction

Soft robotics is a field of research inspired by soft bodied organisms, where engineering and designing aspects of soft structures are the center of interest. Soft robotics can make the interaction between robots and living organisms safe. In addition, soft robots have the potential to function in more natural and complex environments, where rigid robots have disadvantages due to their solid parts. Actuated soft materials, that react to environmental changes add complexity to the design-phase, since the infinite degrees of freedom of soft structures and the possible distributions of materials, make the number of possible designs vast. Therefore, it is certain that designers and engineers, being inspired by nature will stick with a subset of designs, while there will never be enough exploration to the design space, since the approach of such deep design spaces by humans, is a heavy task.

Evolutionary methods can approach such design optimization problem tasks. Solutions, in this case designs, can be represented into the machine with some forms of encoding. Encoding is an essential part of every evolutionary method. Generative encoding has shown promising results especially in specific problem domains, such as evolving controllers for robot gait and morphology evolution. Direct encoding provides a straightforward mapping from genotype to phenotype level; Generative (indirect) encoding determines a set of rules, functions that can be queried and generate each individual solution in the space of the phenotype. Recent work (Cheney et al., 2013) has proved that evolutionary methods coupled with a generative encoding genotype representation can evolve both the morphology and the locomotion behavior of soft robotics in a virtual simulation environment.

Traditional evolutionary methods in pursuance of the objective function defined by algorithm designers, are unable to generate enough diversity within the population driving the evolution towards local optima. *Novelty* search, unlike traditional optimization

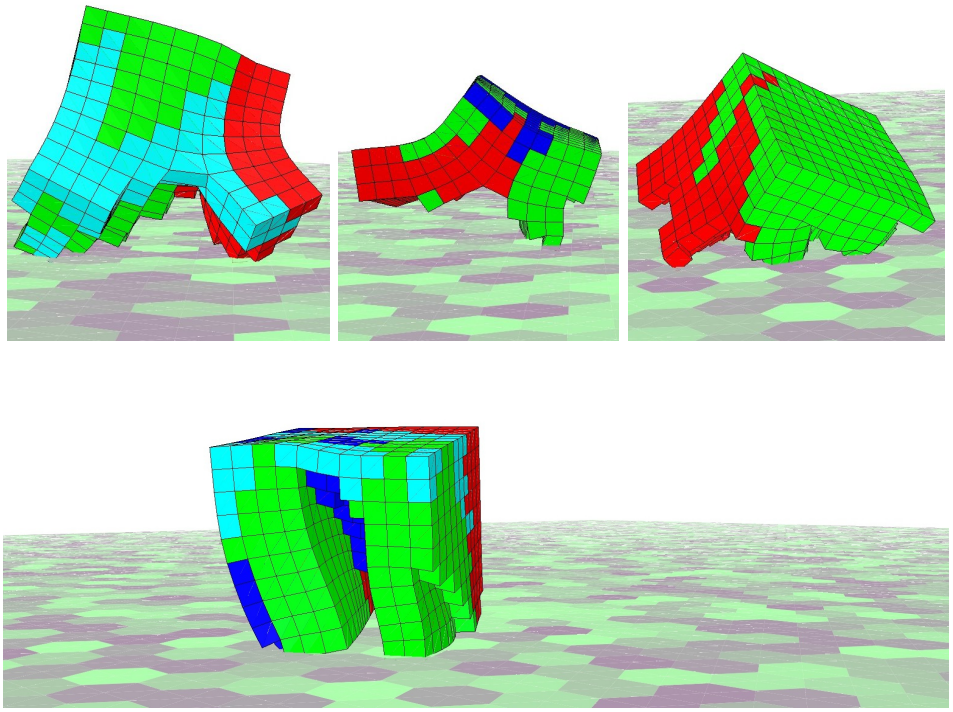


FIGURE 1.1: Different types of morphologies capable of efficient locomotion evolved within novelty and traditional fitness based search.

methods, does not aim to optimize individuals towards an objective. Novelty search rewards diversity and leads to a boundless variety of solutions, mimicking natural evolution. Doing so, novelty search has proven to be a successful method for searching vast spaces where the objective function is deceptive.

Having said about the limitless morphology solutions soft robots can have, it is of interest to investigate kinds of solutions an evolutionary method will evolve. Different environmental variables, such as gravity acceleration, can be decisive as far as the evolved morphologies are concerned. The morphologies as well as the locomotion behaviors that evolved soft robots will acquire during the evolution is a major research question of this thesis, as it can lead to a taxonomy of different morphologies and locomotion strategies for variant environmental conditions. Figure 1.1 illustrates four soft body virtual creatures evolved to locomote under the simulated environment. A variety of morphologies evolved can generate highly efficient locomotion strategies.

1.1 Thesis Contribution

This thesis explores possible ways of evolving the morphology and the locomotion strategy of soft structures in a virtual simulation environment. A random morphology generator is created as a simplistic approach to design soft robots in the specific environment,

resulting in inefficient locomotion ability of the designed structures. The idea of direct and indirect encoding is used in this random framework, to show that a function is a better way to generate morphologies than assigning random shapes to soft robots. Symmetry property is also used in the indirect approach, resulting in more efficient locomotion. An initial experiment is performed to confirm that these problems cannot be captured by a simple genetic method with direct encoding representation of the genotype. To establish a baseline, generative encoding scheme is paired with an evolutionary algorithm to support the claim of previous work (Cheney et al., 2013) that a generative representation can be beneficial in this optimization setting. For the first time novelty search a diversity based method is used for the evolution of the morphology and the locomotion of virtual soft robotics. This thesis is exploring the effect that diversity based search can have on the performance of the locomotion that evolved morphologies achieve. Additionally, it is expected that the diversity of morphologies will be increased under the same settings. Morphologies and locomotion strategies evolved by novelty search show that not only same or better performance can be achieved through this method but also the diversity of behaviors is remarkably increased. A lot of aspects of novelty search are investigated, in an attempt to understand for what reason a diversity based method is performing better than traditional ways of optimization such as fitness based search. An alternative way of selection known as competition is successfully applied to research ways of improving both search techniques. Another contribution of this thesis is a proposed method of incorporating fitness information within novelty search achieving in a considerable improvement to the effectiveness of evolved locomotion strategies. Last, both search methods are used to evolve structures for a variety of gravity levels, expecting to show a different taxonomy of locomotion patterns under different conditions. The effect of gravity in the locomotion velocity of mobile machines is also studied.

1.2 Thesis Outline

Chapter 1 introduces the problem domain this thesis investigates and defines the research questions answered in this work. Chapter 2 provides an introduction to genetic algorithms, different encoding techniques for the genotype representation, neuroevolution algorithms and objective driven search are presented and compared to a diversity based technique, known as novelty search. It also provides insights on the field of soft robotics and some of its applications. In Chapter 3 related material about evolutionary techniques used to evolve artificial life, as well as the evolution of soft robots morphology and locomotion are presented. Chapter 4 is a comprehensive documentation presenting details of the implementation of different evolutionary techniques used. Chapter 5

gives a detailed presentation of the results achieved under different experimental setups. Chapter 6 serves as an epilogue to this thesis where the impacts of the contributions are discussed. In addition, future extensions of this work are provided.

Chapter 2

Background

This chapter gives an overview of the state-of-the-art methods in evolutionary algorithms. It gives an in-depth discussion about the intersection of evolutionary algorithms and robotics. This discussion focuses mostly on how evolutionary methods are used to evolve robot designs and controllers for some applications. In addition, genetic algorithms, the role of the encoding in the representation within an evolutionary setting, how artificial neural networks (ANNs) can represent an organism in evolutionary algorithms (EAs), and how these ANNs can be evolved when coupled with an EA are presented. As part of the different encoding schemes, an indirect coding called compositional pattern-producing networks is also discussed in detail. Additionally, the aspect of the objective function in such evolutionary problems and the effect it has on the performance of the methods is studied. Furthermore, novelty search, a method which uses an objective function that rewards diversity in the evolution is presented in detail. Last but not least, the field of soft robotics is introduced, in conjunction with ways where these soft material structures can be evolved and simulated in virtual simulation environments. Soft robots, designed for real life applications are also presented.

2.1 Evolutionary Algorithms

Evolutionary algorithms (EA) are a part of the evolutionary computation field where generic population-based optimization algorithms are studied. Initially, an evolutionary process holds a fixed number of solutions which are randomly generated. These candidate solutions are propagated within generations until a good solution is found or a maximum number of iterations has passed. One of the most important advantages of EAs is that they can approximate good solutions in very complex optimization problems,

where analytic methods cannot be applied. The non-deterministic nature of evolutionary algorithms starting with candidate solutions randomly sampled from the solution space does not guarantee that the evolution will always come up with the same results on independent runs. Another important fact of EAs is that holding a population of solutions can help avoid being "trapped" in local optima of a specific function. The way EAs propagate from one generation to the other is simply by using all or part of the current candidate solutions to produce the next generation. In evolutionary algorithms, the objective function is the measure that all solutions are evaluated against in order to reach the ultimate goal of the optimization problem.

2.1.1 Genetic Algorithms

Genetic algorithms are part of the evolutionary algorithms following the same principles.

> "Genetic algorithms are probabilistic search procedures designed to work on large problem spaces involving states that can be represented by strings."

Considering the above quote (Goldberg and Holland, 1988) a genetic algorithm is a process of evolving a string-stream of values, which is a single solution in a high dimensional problem space. These values can be at their simplest form bits $(0, 1)$, integers, floats or char values.

Each of these candidate solutions is called a *phenotype* and the stream from which the solution is derived, *genotype* or *chromosome*. Each generation holds a population of a fixed number of individuals which are initially randomly selected out of a distribution over the solution space. The iterative process that follows and creates a new population of individuals, given the current population, is called *generation*. Usually the algorithm terminates after a fixed number of generations or when the goal has been reached.

The way the next generation's population is produced depends on the current population; Genotypes are selected to breed new individuals. There are two basic ways for a new genotype to be produced. The first way is called *mutation* and requires only one individual from the current population. Mutation will change one or more values in the *chromosome* of the selected individual to create a new one and maintain the genetic diversity from one generation to the other. *Crossover* is the second basic genetic operator and requires two or more parents for each new individual. This operator is similar to biological crossover and it uses parts from all parents to create a new chromosome.

The way individuals are selected after each successful generation in order to produce new individuals belongs to the genetic process of *selection*. Selection as the name reveals

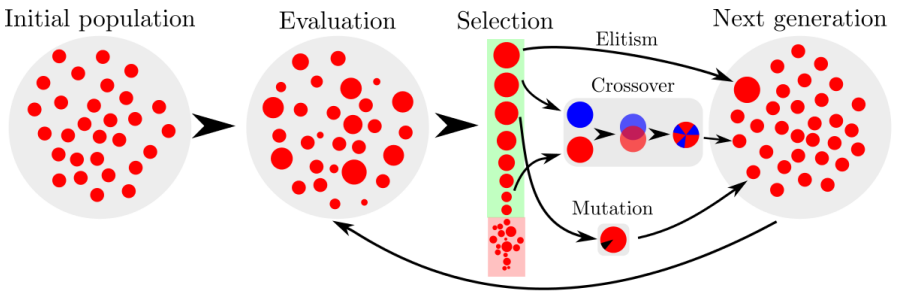


FIGURE 2.1: Basic pipeline of an evolutionary method.

itself selects which individuals will become parents and which individuals will not. The selection criteria as it also happens in some natural environments where the fittest organisms survive is a function that can approximate the goodness of a given solution. This *objective* function, also called *fitness*, is a measure of how good an individual is, i.e. total displacement of a robot's body while trying to evolve walking. With the knowledge of the fitness function added to the evolution, weak individuals are most of the times discarded from the breeding process. Selecting parents randomly from the top part of the population or selecting parents via *tournament*, are two of the basic selection methods in evolutionary algorithms. The former ensures that only a small part (i.e top 20% (*survival threshold*)) of the current population will survive. In the contrary, tournament selection also known as *Competition*, allows the whole population to breed, while it randomly picks a fixed number of individuals selecting the best among those. A third way of choosing individuals for the next generation is called *Elitism*. Elitism is a genetic selection technique. When used, it is responsible for copying a mutation or an actual copy of the best individual of the current generation to the next. It ensures that during the evolution successful solutions will carry on living and share their "valuable" genes into the next generations.

Figure 2.1 illustrates the general algorithmic pipeline of an evolutionary method, as described above. This starts with a random initialized population which is then evaluated (size refers to how "good" each individual is). All individuals are then sorted based on their goodness in respect the objective function. The selection process follows, where a set of the best individuals is selected to produce the next generation. Elitism, alongside crossover and mutation are used to this proportion of the population. The next generation will also be evaluated in respect to the same objective function and the iterative process will continue.

2.2 Evolutionary Robotics

Evolutionary robotics (Nolfi and Floreano, 2001) (ER) is a method that makes use of evolutionary computation algorithms to evolve the control and/or the morphology, without the direct design by engineers. Most research concentrates on developing robot controllers for simulated or real robots (Harvey et al., 1997; Nolfi et al., 1994). One big advantage of this method is that it can evolve solutions for environments that designers and engineers do not have enough knowledge about (i.e., designing a robot controller for another planet, where surface type and gravity level might be crucial variables for the design of an exploring robot). In the same fashion as natural evolution, evolutionary techniques work with a population of random initialized controllers or designs. The candidate population individuals (robot controllers) used in ER applications may be drawn from some subset of the set of artificial neural networks (ANNs), whereas simpler versions of genetic algorithm applications use bit-streams that directly map the controller. The controllers in the best performing robots are then selected, altered and propagated through mutation, crossover, and other genetic operations, in a repeating process that mimics natural evolution. Evolutionary robotics is done with many different objectives, often at the same time. These include creating useful controllers for real-world robot tasks, reproducing biological phenomena, etc.. Creating controllers via artificial evolution requires a large number of evaluations of a large population. This usually takes a lot of computational time, which is one of the reasons why evolution of such controllers is usually evaluated within a simulation software. Initial random controllers may exhibit potentially harmful behavior, such as repeatedly crashing the robot into a wall, which may damage a physical robot.

Apart from evolutionary methods to develop robot controllers reinforcement learning (Hayes and Demiris, 1994; Mahadevan and Connell, 1992) can be used rewarding actions, resulting to state-action pairs that lead to high rewarding behaviors. As a result, a robot controller can be indirectly built. Applying evolved robot controllers to real robots in a physical environment is an extremely difficult task, since simulators in front of the limitations of computing efficiency sacrifice the accuracy (Jakobi et al., 1995). As mentioned earlier, evolutionary methods can be used to design the physical structure (morphology) of a robot (Hiller and Lipson, 2010), in addition to or in place of the controller. This thesis is exploring this aspect of evolution, the simultaneous evolution of the morphology and the locomotion of virtual soft robots.

Developmental robotics (Lungarella et al., 2003; Asada et al., 2001; Weng, 2004; Asada et al., 2009) is a field related to evolutionary robotics, while instead of evolving through generations towards fitter controllers, it is trying to mimic life-like learning starting

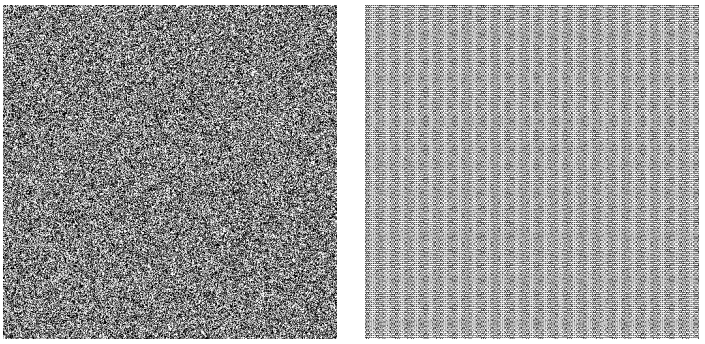


FIGURE 2.2: Comparison of direct encoding versus generative for the binary image example.

from a "blank" state in which the robot's "brain" is initialized and every variable of the environment is unknown.

2.3 Direct-Indirect Encoding of the Genotype

A simple direct encoding was described in the previous section, when a single dimension stream of bits or numbers described the chromosome. When the dimensions of the task define the length of the genome, we refer to *direct* encoding, which means that the genotype-phenotype mapping is a straightforward function. An example of this encoding could be the design of a two dimensional binary image. In direct encoding the genotype of this picture can be represented by a stream of bits which has the same length as the number of pixels of the image. In other cases, where there is no direct mapping between the genotype and the phenotype, indirect encoding is present, where a set of rules or a function maps the genotype to the phenotype space. In cases the phenotype space can be represented by a Cartesian n-dimensional space, an indirect encoded chromosome can be a function that is queried for each coordinate in a specific resolution and represents the phenotype. For the same binary image example, indirect encoding would be a function that gives pixel values 0 or 1 for every pixel's coordinate.

Figure 2.2 illustrates the difference between direct and indirect encoding. An example binary image is shown for both encoding schemes, in the first case (direct encoding) the genotype is a binary stream which length is equal to the number of pixels producing the value of each pixel directly. The latter encoding uses a genome of length 3, as many as the coefficients of the linear combination in the following function:

$$f(x, y) = c_1 sin(x) + c_2 cos(x) + c_3 tan(y)$$

the result is taken after applying the same function for each pixel coordinate. Even in cases where a simple function is used, the phenotype holds some of its functions

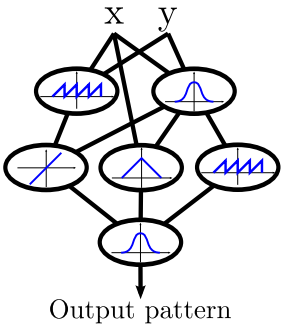


FIGURE 2.3: Compositional pattern-producing networks have identical network structure with artificial neural networks while they make use of a canonical set of activation functions.

properties such as symmetry and repetition, resulting in a pattern that direct encoding cannot produce.

A method that can represent more complex functions and is widely used to indirectly map the genotype to the phenotype space is the *artificial neural networks*. Artificial neural networks (ANNs) are computational models which are inspired by living-organisms central neural system (Brain). These models are used to approximate functions that are generally unknown, using a set of nodes and connections between pairs of nodes. Each connection within the network holds a weight which used as a multiplicative factor of each signal passing through the connection. Nodes are then responsible of propagating the summation of the signal received from the connections by a Sigmoid function. This interconnected set of nodes can propagate the inputs fed into the network to one or more output nodes, approximating in this way a complex non-linear function.

2.3.1 Compositional Pattern-Producing Networks

Encoding plays an important role and it is critical to the performance of evolutionary algorithms especially when large problem spaces are present. Research has shown that the genotype-phenotype mapping can affect performance (Komosiński and Rotaru-Varga, 2001) in three dimensional agents, where more complex encoding schemes outperform direct encoding. In addition, geometrical implications of the problem also have some potentially important roles in the encoding. The role of symmetry to the encoding is crucial especially in applications like board games, robot controllers, biped walking, etc.. In these cases, geometric regularities of the encoding can be essential to the performance of the evolutionary method.

Compositional pattern-producing networks (Stanley, 2007) or CPPNs are artificial neural networks with an extended set of activation functions (see Fig. 2.3). Results by this encoding show that regular patterns can be produced in this generative mapping from

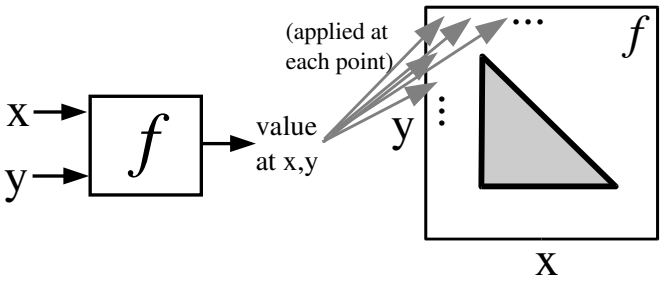


FIGURE 2.4: CPPNs work as a function f that is being queried for the whole n-dimensional Cartesian space in which space the phenotype is mapped, in this case the phenotype is the triangle in a two-dimensional space, figure taken by (Stanley, 2007).

the genotype to the phenotype space. Like in the previous two dimensional image representation of a phenotype, CPPNs generate phenotypes that can be interpreted as distributions of points in a multidimensional Cartesian space. The genotype (CPPN) can then be queried for each coordinate of the space and gives the phenotype representation of the genotype in multiple resolutions. In the same fashion, images can be constructed using CPPNs, where pixel coordinates are queried to the network and the grayscale or RGB values can be taken by the outputs of these networks.

Figure 2.4 illustrates how the mapping between the genotype and phenotype is done using generative encoding (CPPNs). A major asset of CPPNs is that they can generalize in all resolutions. Considering the previous figure (see Fig. 2.4), the CPPN is queried for all x, y coordinates of the phenotype two dimensional Cartesian space. The step of x, y sampling can be determined by the problem, since the inputs of the CPPN are the normalized coordinates $x, y \in [-1, 1]$. Hence, genotypes using this kind of generative encoding can be mapped in every resolution, making this process straighforward to generalize. As the space of the phenotype becomes larger, a generative encoded solution (CPPN) is not affected by the increasing dimensions of the problem, a constraint that heavily affects direct encoding.

Compositional pattern-producing networks have been used in many applications where symmetry and repetition can produce two or three dimensional artistic structures[2], and drawings[1] (Secretan et al., 2008). As these applications require more symmetrical properties than others, not only Cartesian space coordinates are fed into the inputs of these networks, but more inputs biasing the network should be present (Secretan et al., 2008). Some example inputs that can be fed into the network as additional inputs are the distance from the center of the space or the distance from the center to one axis. Figure 2.5 illustrates images encoded by CPPNs. Comparing the results with Figure 2.2,

[1] picbreederSite: `http://www.picbreeder.org`
[2] EndlessForms: `http://www.endlessforms.com`

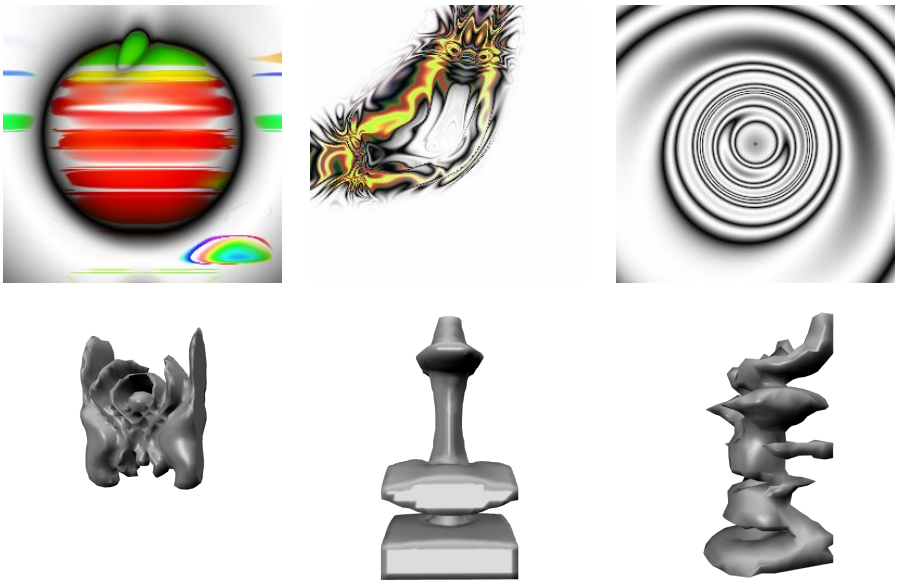


FIGURE 2.5: Compositional pattern-producing networks can encode truly complex images[1] (top) and 3D-structures[2] (bottom).

it is understandable why this kind of encoding can capture solutions in problem domains where symmetry is important.

2.4 Neuroevolution

Neuroevolution (Yao and Liu, 1997) is an optimization technique using evolutionary methods as described in Section 2.2, where artificial neural networks take the place of simpler encoding methods. ANNs can compute arbitrarily complex functions, learn and perform under the presence of noisy inputs and generalize to unseen sensory information. Neuroevolution requires only a measure of a network's performance at a task, which can be used as the reward for good solutions (ANNs) to survive. More complicated forms of chromosome representations can develop more complex robot controllers. After each run, the sensory input of the task domain is given at the artificial neural network's input neurons and the solution is given by the output of the networks where the fitness of the specific brain can be evaluated. A major issue is the selection of the network's *topology*. Topology is the arrangement of the network's elements such as links and nodes, which represents the structure and how the information flows within the network. In early neuroevolution methods the topology of the networks used was fixed, meaning that the only elements of the networks evolving were the weights of the connections between the nodes. In modern neuroevolution methods, the topology of the networks is also subject to the evolution.

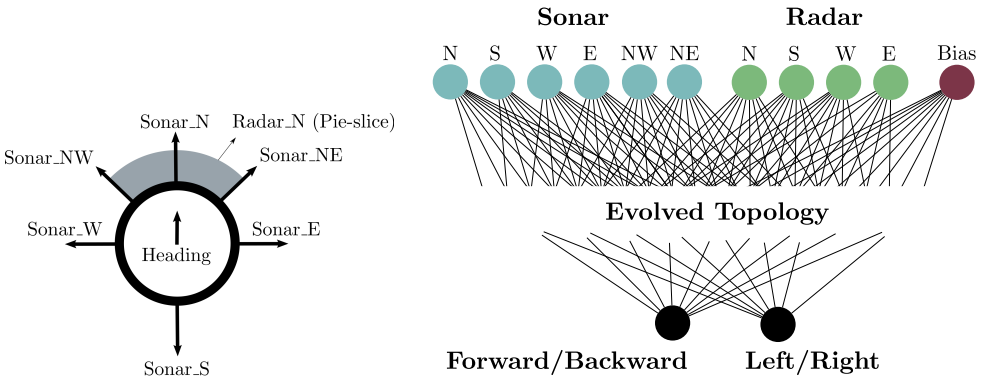


FIGURE 2.6: Robot controllers can be evolved through neuroevolution algorithms where robot sensors are the inputs of neural networks while the outputs directly control the robot.

2.4.1 Neuroevolution of Augmented Topologies

Neuroevolution of augmented topologies (NEAT) as it was first introduced by (Stanley and Miikkulainen, 2002) is also a neuroevolution method used to evolve artificial neural networks. A major advantage of this method is that alongside the weights it also evolves the topologies of the networks within the population.

Originally, neuroevolution methods were developed to capture difficult sequential decision making, as well as to control problems. The sensory information is the input of these neural networks and decisions are the outputs. NEAT is yet another method for evolving ANNs where a few extra features are added, enables finding solutions in more demanding problems. NEAT starts the evolution process with a population of networks with simple topologies. Through the generations instead of just fixing the weights of the networks' connections, topologies are becoming more complex allowing nodes and links to be added. Meaning that during the evolution, more complicated networks will be produced, this *complexifying* technique leads to capturing more demanding solutions as it offers enough freedom to the evolution.

Figure 2.6 illustrates how sensory information can be given as input to a neural network. The neural network, given the sensory information provided, controls the robot which tries to drive itself close to a target position in a maze. The outputs of the network completely control the motion of the robot. All the sensory information (six sonar sensors which output the distance from the closest obstacle in six directions and 4 pie-slice radar sensors which are only activated when the target position is located within the range of each one covers) is available to the controller.

Several aspects of this method worth mentioning where *speciation* is one of the most important. Speciation is the procedure that protects new *species* until they have enough time to evolve before comparing them with the rest of the population. For two individual

genotypes (ANNs) to belong to the same species their network topology must be similar, meaning that a threshold is set and a function determines the numeric value of two network topologies' similarity. Two different genotypes (ANNs) can share shame genes (topologies within the network) and their compatibility (genotype similarity) is given by:

$$\delta = c_1 \frac{E}{N} + c_2 \frac{D}{N} + c_3 \bar{W} \tag{2.1}$$

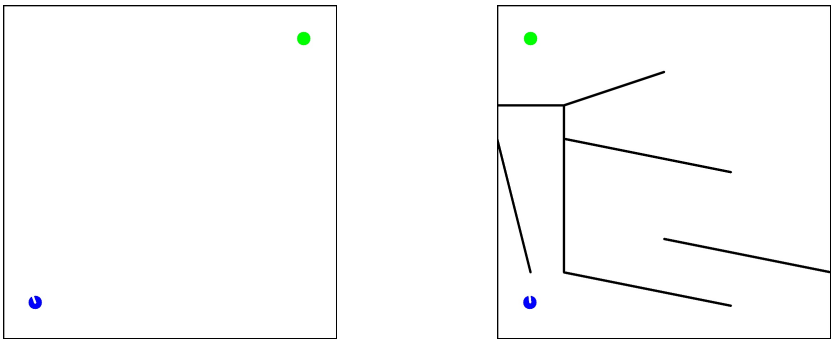
where E is the number of *Excess* genes (genes that do not match and they do not occur in parents' genotype), D is the number of *Disjoint* genes (genes that do not match but they occur in parents' genotype), $\bar{W}$ is the average weight difference between *Matching* genes (Identical) and N is the number of genes in the larger genotype used for normalization. The age of each species protects them for competing in equal terms with more optimized species, giving them in this way time to evolve further towards the objective function.

2.5 CPPN-NEAT

Compositional pattern-producing networks as described earlier in this chapter (see Sec. 2.3.1) are similar computational methods to ANNs in regards to their structure, so one can make use of the *complexifying* property to capture in this way more complex solutions (behaviors). NEAT method can evolve CPPNs in the place of ANNs, since it only needs few modifications.

The resulted method that evolves this generative type of genomes (CPPNs) is called CPPN-NEAT (Stanley, 2007) and its only difference in respect to the original NEAT algorithm is the way new nodes are added to the network. The original NEAT algorithm evolves ANNs which are using sigmoid functions at every node, so every new node will carry this function. In the contrary, CPPNs use a variety of functions from a canonical set. CPPN-NEAT assigns a random function from this set to every newly added node.

Experiments (Stanley, 2007) have shown that this method can indeed evolve CPPNs capturing in this way solutions in problems with geometrical properties (i.e board games, biped walking, etc.). NEAT is holding the properties of natural evolution as every neuroevolution method. Furthermore, NEAT coupled CPPN encoding can be used to determine the connectivity (topology) of artificial neural networks in a method called HyperNEAT (Stanley et al., 2009).



(A) Easy (B) Hard

FIGURE 2.7: An objective function can be devious. Maze examples from (Lehman and Stanley, 2011a).

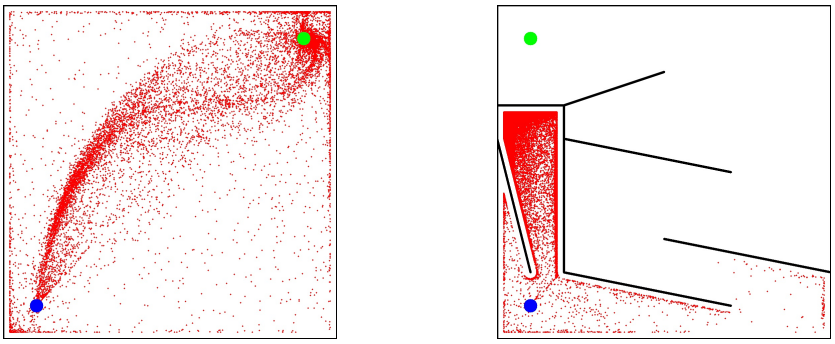
2.6 Novelty Search

Traditional search within the framework of evolutionary algorithms needs an objective function, a function that guides the search towards "good" areas of the solution space following the gradient of the fitness. Defining the fitness function is a straightforward problem most of the time. In a problem where a robot tries to get to a target from its initial position in a room with no obstacles in between a fitness function could be defined as the Euclidean distance between the final position of the robot and the target point, the closer it gets to the target the more points (higher fitness) the specific controller is rewarded.

When the objective function misleads the search

An objective function as the one described above is greedy, driving the search directly towards highly rewarding areas of the solution space. In cases that local optima can be found in the landscape of the objective function this greedy fitness measure can drive and trap the evolution in these localities of the problem.

Considering the robot-maze example presented in (Lehman and Stanley, 2011a, 2010), a robot (blue dot) is placed in a maze (see Fig. 2.7), the robot (see Fig. 2.6) has multiple sensory information which are fed as inputs to its controller ("brain"). The controller is driving the robot through the maze having only sonar and radar sensory information, while its ultimate goal is to drive the robot to the target position (green dot) in a fixed time span. Naturally, to select a fitness function that can give enough information about how good a controller is the Euclidean distance from the final position of the robot to the target position is measured in the end of the simulation time. For the first maze (see Fig. 2.7a) when no obstacles are between the robot and its target the objective function is reliable, since the Euclidean distance to the target indeed informs the robot how close



(A) Visited space after 200 generations.

(B) Visited space after 1000 generations.

FIGURE 2.8: Fitness search has no problems to find the solution in the easy map, while it can not find the optimal solution in the hard setting after 250.000 evaluations.

it is located. In the second maze (see Fig. 2.7b) using the same fitness function search can be mislead. In this example maze achieving high fitness does not mean that the robot is actually close to the target. Driving north in this maze following the increasing fitness leads to a wall that cannot be passed by the robot. Therefore, exploration is needed in low fitness areas which will allow the robot to reach the target point with the maximum fitness. The deceptive nature of the fitness function in this problem can be found in a lot of optimization problems, while the walls in this maze clearly denote problems where this fitness landscape can be found.

To visualize how fitness based search can fail in such a setting Figure 2.8 presents the results of the above experiment explained using the robot sensory information presented in Figure 2.6. NEAT algorithm is used to evolve the neural controller presented in the same figure. The settings used for this experiment were the same as in (Lehman and Stanley, 2011a) where a population of 250 individual controllers per generation was used. As it was expected, fitness based search was successful in the easy setting (see Fig. 2.8a). However, it failed to find the optimal solution in the hard map (see Fig. 2.8b) focusing on creating controllers that lead the robots drive north until the wall was reached, failing to explore the map extensively.

Natural evolution is not an evolution towards fitness

Using an objective function in evolutionary computation and typically reward individuals which are closer to an objective is far away from natural selection in the evolution process, where exploration is allowed as long as the criteria for survival hold (Lehman and Stanley, 2010). Driving search towards promising parts of the fitness space where local optima may be present ensures that other areas of the search will not be explored, leading search to stay and explore the nearby area while more promising regions are

far away in the solution space. Solutions located in these regions are called *stepping stones* (Lehman and Stanley, 2011a, 2010, 2008; Risi et al., 2009). Stepping stones are points in the solution space that may not be good as far as their objective values are concerned, but can eventually lead to better or the global optima of a specific optimization problem.

Search for novelty

Novelty search (Lehman and Stanley, 2011a, 2010, 2008; Risi et al., 2009) unlike traditional fitness based search is an alternative way of optimization towards an objective function without having knowledge of this objective. In simple words it is looking for a solution to a problem without knowing how close it is to solve it; fact that turns out to have a major impact to the increased performance of this method in several problem domains.

What novelty search seeks for is how interesting a new solution is in respect to all previously found ones. To define "interesting" we need to move our point of interest into behavior space which is a function of each phenotype, similar to the fitness function. Nevertheless, it fully or partially describes the behavior without directly implying the fitness function. As an example someone can think of a behavior could be defined as the final position of the navigation robot or the trajectory of it in the previous robot-maze example. Rewarding behaviors of the phenotype that are different from the previously found ones drives the evolution to visit new points in the behavior search space.

One significant point here is that the behavior space in some domains can be limitless. However, a valid behavioral metric can be found excluding behaviors that are meaningless or do not comply with the natural limits of the problem. On the other hand, the search space in the genotype level can also be infinite especially in neuroevolution methods like NEAT where ANNs can grow during the evolution. A bounded space of understandable-valid behaviors is then the key idea of novelty search where increasingly complex behaviors present to the evolution as the complexity of the genotype grows along.

Multi-objective optimization can also make use of a novelty metric alongside fitness, trying to optimize both at the same time (Mouret, 2011). Another method that exploits the diversity of the produced genomes in order to map the phenotype to the fitness is also proposed by the literature (Mouret and Clune, 2012).

Is novelty search similar to random?

Initial thoughts are converging that novelty search is similarly behaving to random search, constantly looking for something new in the vast space of behaviors. It seems similar to evolving random robot controllers without considering the behavior aspect of their phenotypes hoping that enough exploration will be done in both genotype and phenotype spaces. A random approach having no information about the observed behaviors the evolved phenotypes produce is not able to drive the evolution since different and more complex genotypes can easily produce similar behaviors. The novelty in the behavior level assures that the search will explore deeply the behavior space with the hope that a *fit* behavior will be found. Aside from that, novelty search does not perform backtracking which ensures that it will constantly drift away from already generated behaviors (i.e similar behaviors to already generated ones result to low novelty value). At the same time there is no such guarantee in random search. Therefore, it is certain and proven later in this thesis that no exploration in the behavior space will be performed by random search.

How can novelty be measured?

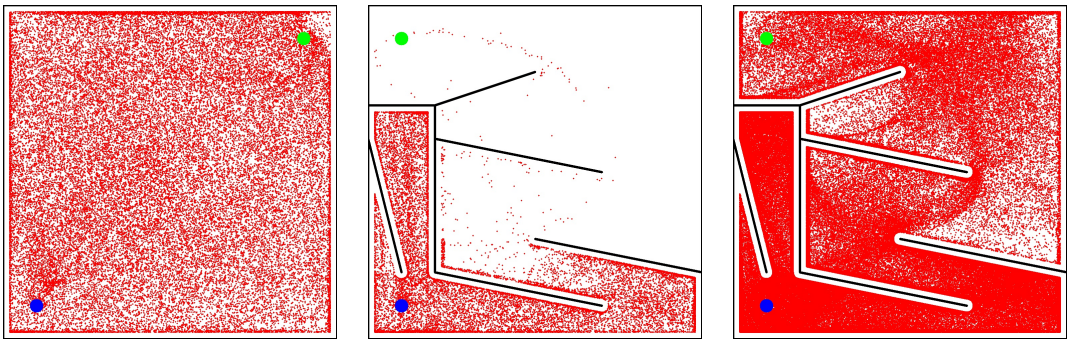
As fitness is a function to measure the "goodness" of an individual, novelty measures how different an individual is from all previously found individuals. To define different a novelty metric measures the difference in the behavior space of the phenotype. Given the phenotype's behavior x a novelty measurement could be a function of x, $f(x)$ which computes how different (novel) is the specific behavior in respect to a set of other behaviors S in behavior space. As defined in (Lehman and Stanley, 2011a, 2008) *sparseness* can give a good measurement of how sparse is the area of a newly observed behavior. Given the behavior we can compute the sparseness by:

$$f(x) = \frac{1}{k}\sum_{i=1}^{k} dist(x, S_i) \tag{2.2}$$

where S is a sorted set of the closest behaviors. Sparsity measures the average distance from the k-closest behaviors.

Algorithm

Replacing fitness with a novelty value is not the only modification any evolutionary algorithm needs in order novelty search to be implemented. To push search to visit new areas in the behavior space rewarding novel behaviors coming up during the evolution is



(A) Visited positions in the map after 200 generations.

(B) Solution found after only 80 generations.

(C) Visited positions in the map after 1000 generations.

FIGURE 2.9: Novelty search applied in the robot-maze optimization problem. Novelty search deeply investigates the behavior space finding the solution even in the hard map setting.

needed. For this reason, storing novel reference points in space (behaviors) during the evolution is inevitable. The sparseness of a new behavior is computed by Equation 2.2 resulting in a numerical value that implies how novel is the observed behavior of an individual phenotype. If the new behavior has a novelty value more than this threshold it is stored in the set of novel individuals. Apart from comparing any new behavior with all the novel behaviors, the newly produced one can also be confronted with the entire set of behaviors produced by the population in the same generation of the evolution.

Having discussed the basic idea behind novelty search and how it can be implemented, it is time to apply it in a known problem where fitness based search failed. Considering the robot (see Fig. 2.6) - maze (see Fig. 2.7) example presented in this section, novelty search is now taking the place of evolution towards the objective function used before which was the distance to the goal. For the novelty metric to be evaluated, a behavior metric has to be defined, which in this case can be the final position of the robot by the time the simulation is finished. The sparsity measure then computes the reward of the robot based on how sparse the observed behavior of the robot in regards to all novel behaviors found before in the evolution is, based on the sparsity equation (see Eq. 2.2) using $k = 10$. Figure 2.9 presents the results achieved by novelty search in this setting by showing all the visited areas that robots were driven to by their evolved controllers. In the easy setting map (see Fig. 2.9a), novelty search achieved to fully explore every possible position in the map. In the hard map (see Fig. 2.9b, 2.9c), where fitness based search failed to find any solutions close to the target position, novelty search succeeded to do so after only 80 generations.

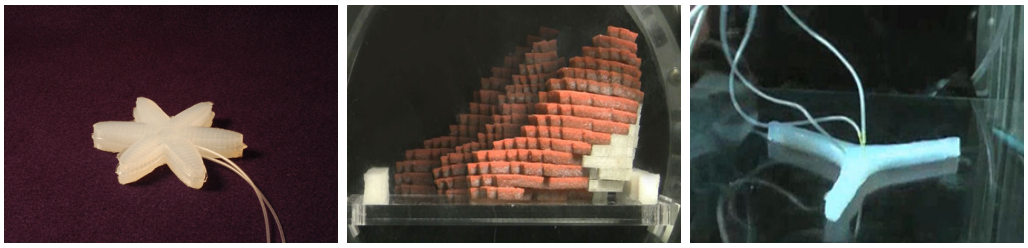


FIGURE 2.10: Soft robots can be actuated through air pressure tubes (left), pressure variations (middle), or internal explosions (right).

2.7 Soft Robotics

Soft robotics is a highly promising field of research dedicated to the science and engineering of soft materials in mobile machines. As the name suggests soft robots (Trivedi et al., 2008; Pfeifer et al., 2012) are made entirely of soft materials mimicking animals or animal-parts that consist only of soft tissue (elephant trunk, tongue, worm, octopus, etc.). Having no rigid parts in their design the degrees of freedom are infinite and the possible ways of motion can become extremely complex. In traditional robotics, joints and rigid parts predefine the space of possible movement and sometimes restrict the robot's locomotion strategy or *gait* to a specific set. In soft robotics, the absence of rigid parts can on the one hand make the design of the locomotion strategy exceptionally tortuous, on the other hand the gait alternatives are limitless.

The design and development of soft robotics is not an easy task, while the actuation of such soft structures is the most challenging task. Actuating soft materials can be done in many ways including pneumatic systems (Ilievski et al., 2011; Shepherd et al., 2011), hydraulic, internal body explosions, passive actuation triggered by pressure or temperature variations and others (Laschi et al., 2012; Seok et al., 2010). Figure 2.10 illustrates three different ways that soft robot bodies can be actuated. Gripping mechanisms (Hirose and Umetani, 1978) can softly and gently conform to objects of any shape and hold them with uniform pressure. This gripping function is realized by means of a mechanism consisting of links and series of pulleys which can be simply actuated by wires. Regardless traditional ways of actuating soft material robots, three dimensional printing is now giving the freedom for multi-material structures to be created, which also explodes the number of possibilities for the design of a soft structure such as a gripper soft robot. Topological optimization techniques can be applied (Hiller and Lipson, 2009) for producing functionalities in the design. Autonomously actuated soft robots (Tolley et al.) (see Fig. 2.11) can also be designed having multiple advantages over rigid body robots such as resistance under extreme temperatures and the capability of locomotion on terrains of variant types.

FIGURE 2.11: Autonomously actuated soft robot (Tolley et al.), it is able to withstand extreme temperatures and variant terrain types.

Although soft robotics research field is in an early stage, it is growing fast. Some of the characteristics that make soft robots interesting to explore are the infinite number of degrees of freedom and the variety of materials (mostly elastic) that can be used, in the contrary to rigid robotics that are mostly made out of metals and plastic. Nevertheless, structure design and control of soft robotics remain challenging mostly because of their soft bodies can only be represented in continuous state spaces, where only analytic methods can be proven successful.

To sum up, locomotion capabilities of soft robots, as well as the possibilities of passive movement (i.e materials that actuate reacting to environmental changes) makes them an interesting topic for present and future research. Finally, considering also that soft materials are safer than conventional robot materials to humans, human-robot interaction can benefit from this field (Sanan et al., 2011).

2.7.1 Soft Robotics in Simulation

Most work to simulate interactions and deformations within and between soft material bodies are mostly focused on the graphical part of the problem (Faloutsos et al., 1997) sacrificing the accuracy of the simulation (Teschner et al., 2004). Three dimensional meshes (Müller et al., 2002) can represent these bodies including the dynamics of their materials.

A recent work though, *VoxCad* simulator (Hiller and Lipson, 2012a) is focusing mostly on the physics side of the soft material interactions not at the expense of a low frame rate. *VoxCad* is a modeling and analyzing open-source software that can simulate soft material deformations and interactions. In Figure 2.12 the graphical user interphase

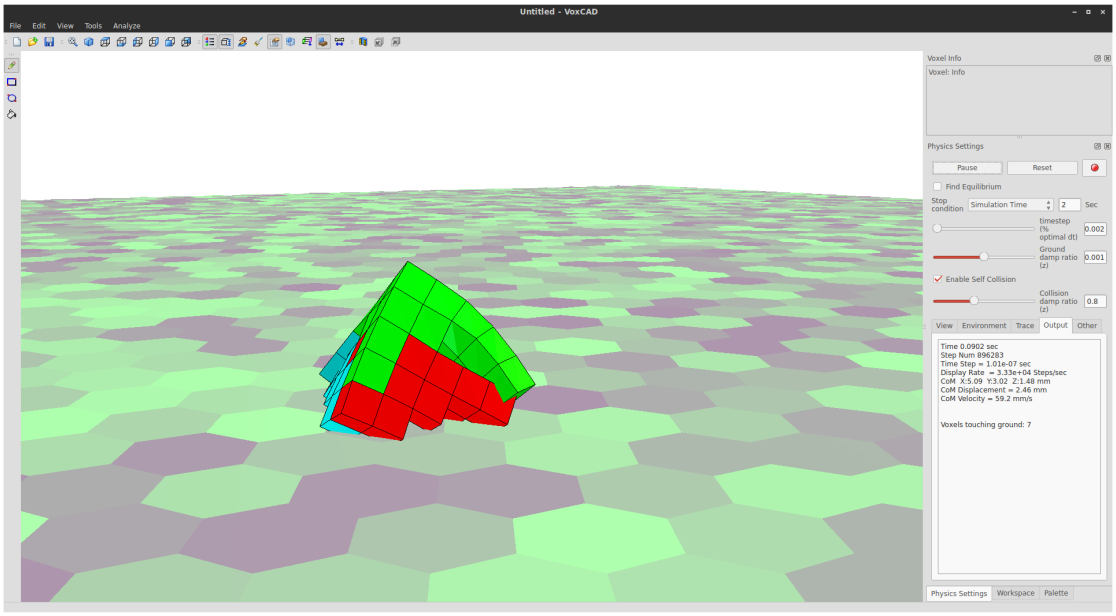


FIGURE 2.12: VoxCAD (Voxel CAD), a cross-platform open source voxel modeling and analyzing software.

of VoxCad software is presented during the simulation of the soft body robot in the simulator.

VoxCad cannot model and simulate three dimensional meshes, yet a lattice is used to represent the 3D workspace where voxels (three dimensional pixels) can be assigned different materials. Materials themselves are passive and cannot actuate without external trigger. In this simulator this external force that can actuate the materials is the temperature of the environment. The main variables of the environment is the base, the amplitude and the period of the temperature. Furthermore, gravity acceleration of the environment can vary. Materials have properties such as the elasticity of the material, density, Poisson's ratio, coefficient of thermal expansion (which determines how materials will be expanded in respect to the environment's temperature), temporal phase in respect to the temperature period, and the ground friction coefficients. Materials can also be mixed together to create a new type of material.

Throughout this thesis, the terms *structure* or *soft robot* will refer to a set of connected voxels (not unconnected parts) within the lattice space. The voxel dimensions are constant through the experiments of this thesis, while the lattice space is variant. Since the voxel dimensions are the same for all settings, the term *resolution* will be referring to the number of voxels in each dimension. Note that different resolutions also refer to different dimensions for the lattice as the voxel size is fixed. For experimental settings used during the simulations see appendices A, B.

Chapter 3

Related Work

This chapter presents related research work in evolutionary robotics and methodologies used to evolve robot controllers as well as robot morphologies in simulated (*Artificial Life*) or physical environments. In addition, a lot of research work has been conducted regarding the aspect of the encoding to the morphological evolution of soft robots. With the design freedom soft materials give to any evolutionary method, it is of interest to see what has been achieved so far. Most work utilizes a fitness based evolution to successfully evolve virtual and physical robots. However, as it will be discussed later in this chapter, novelty search has been used within an evolutionary setting in order to evolve virtual creatures. Novelty search, as it was discussed in Section 2.6, is a diversity based method where the objective function rewards the novelty in the behavior level.

3.1 Evolution of Virtual-Physical Robots

Robot controllers can be evolved through evolutionary algorithms on simulated (virtual) robots. Moreover, evolutionary methods can be applied to physical robots (Nolfi et al., 1994) where no damage can occur due to exploration of the action space. Controllers represented by an encoding scheme can be generated and propagated from generation to generation within an evolutionary framework until good solutions are found.

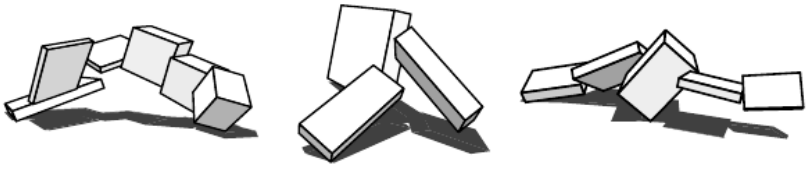


FIGURE 3.1: Karl Sims, *"Evolution of virtual creatures"* (Sims, 1994).

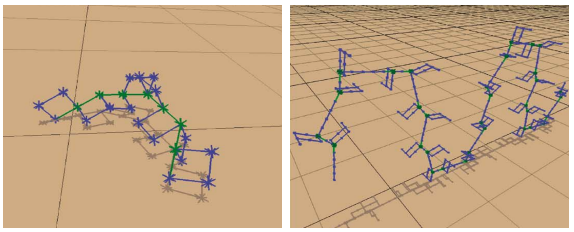


FIGURE 3.2: The use of Lindenmayer systems results in creature morphologies that have a more natural look (Hornby and Pollack, 2001).

Novel systems that make use of evolutionary methods to evolve complex encoding representations such as artificial neural networks have been developed. These complex representations can control not only the morphology of rigid body parts connected with joints, but also control the forces applied to each joint. As a result, virtual creatures (see Fig. 3.1) can be produced in a physical three-dimensional world (Sims, 1994). Different fitness measures also give the possibility to the evolution of diverse creatures in respect to these measures. This genetic encoding defines a hyperspace of infinite number of possible creatures and behaviors, when it is searched using optimization techniques like EA a variety of successful and interesting locomotion strategies emerge, some of which would be difficult to invent or build by engineers. This was the first work successfully tried to evolve both the morphology and the locomotion of virtual robots in a simulated environment, based on such a complex representation for the genome (ANNs).

Computer graphic designers can profit from evolutionary techniques since the design phase of some applications (i.e games, movies, etc.) is a time consuming process. However, the need for natural looking morphologies is of crucial importance in such optimization methods. Previous work (Sims, 1994; Lipson and Pollack, 2000) resulted in unnatural looking shapes for the evolved virtual creatures and abnormal behaviors mostly due to the vast solution space and the encoding representation of the genome. A system that uses Lindenmayer systems (Hornby and Pollack, 2001) (L-systems) as the encoding of an EA for creating virtual creatures was proposed. Creatures evolved by this system have hundreds of parts, while the use of an L-system as the encoding resulted in creature morphologies that have a more natural look (see Fig. 3.2). The discussed method (Hornby and Pollack, 2001) showed that the encoding of the genome can indeed have a big impact on the evolved morphologies.

Evolutionary methods have shown the ability to create complex designs for robots which can perform tasks in the environment they are evolved in. However, these complex designs are hard or sometimes impossible to be transferred on a physical robot. Generative representation used in (Hornby et al., 2003), accomplishes to replace complex representations into a construction plan which uses simple robot components in a regular way (see Fig. 3.3). This compact design space of the resulted method can indeed limit the possible

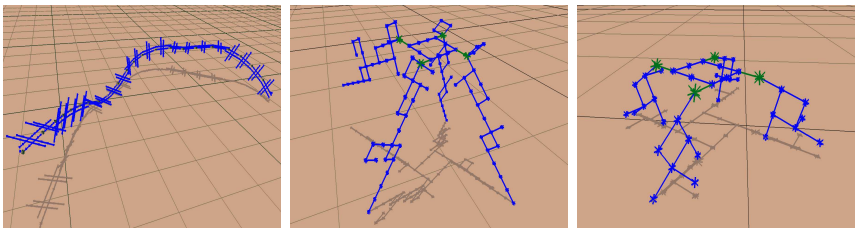


FIGURE 3.3: Generative representation can define a set of rules that simple components can be put together to generate a robot (Hornby et al., 2003).

morphologies given a set of possible morphological parts. As direct encoding schemes have trouble capturing geometrical properties of the problem, generative encoding like CPPNs can be used in order to take advantage of a problem's regularities.

HyperNEAT (Stanley et al., 2009) is a method to evolve CPPNs which then determines the topology and the weights of ANNs. It has shown promising results in evolving the gaits of legged robots (Clune et al., 2009), whereas direct encoding schemes have not been successful. Natural evolution is the only process which instead of evolving only the brain of biological organisms, it also evolves the morphology of them. CPPN-NEAT (Stanley, 2007) can be used as a generative encoding EA which can evolve both features of virtual robots (Auerbach and Bongard, 2010a,b) (see Fig. 3.4), verifying that more complex creatures than designers imagination can be created in such a setting. With this representation it is also possible that a lower resolution phenotype space can be used in the first runs of the evolution to save computational time without significantly degrading the quality of evolved structures, while later a higher resolution space can be used for a more detailed optimization.

Evolving objects with types of encoding based on concepts from biological development like CPPNs can be a powerful way to evolve complex and interesting objects (Clune and Lipson, 2011). These results can be used in applications in fields of engineering, biology, and in others as diverse as art. Apart from the use in robot-bodies design evolution, EA techniques coupled with indirect coding schemes allow the evolution of the morphology and the motion control of soft bodies. In this case multicellular animats (Joachimczak and Wróbel, 2012) in a two-dimensional fluid-like environment. Both the developmental

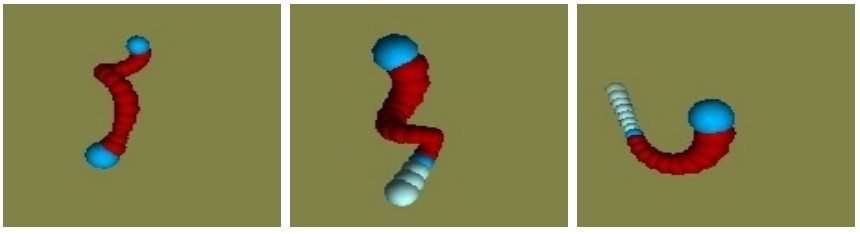


FIGURE 3.4: CPPN-NEAT can be used as a generative encoding for the evolution of virtual robots (Auerbach and Bongard, 2010a).

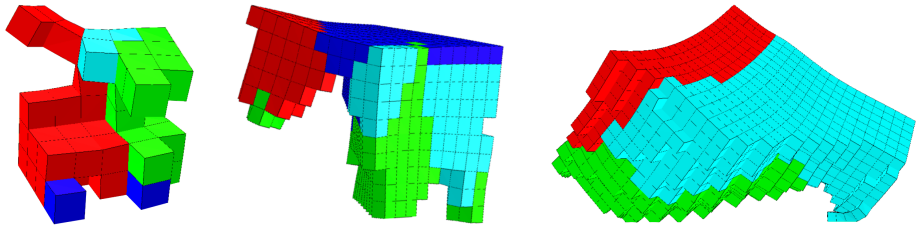


FIGURE 3.5: *"Unshackling Evolution: Evolving Soft Robots with Multiple Materials and a Powerful Generative Encoding"* (Cheney et al., 2013).

program that determines the morphology and the motion control are encoded indirectly in a single linear genome, where a genetic algorithm can be applied to evolve it.

With the excel of 3D printing, soft multi-material robot bodies can be produced using simple material types. These soft structures entirely made of soft-materials can be simulated (Hiller and Lipson, 2012a) allowing the evolution of their designs without the costs of production. As it was first shown in (Hiller and Lipson, 2012b), the automated design of three-dimensional bodies can obtain many functionalities through the distribution of different materials inside their body. The virtual soft robots were successfully evolved (EA) and tested for a single-direction locomotion displacement, while the best evolved morphology was printed into a physical soft robot using a three-dimensional printer. The soft robot tested inside a pressure-chamber and achieved to move itself with a displacement that had only a small error compared to the one in the software simulation.

Evolution of soft material robots as it was shown in (Hiller and Lipson, 2012b), can result in soft robots able to produce locomotion. The possibility of evolving these soft structures using an indirect encoding was of interest to be exploited by (Cheney et al., 2013). A powerful generative encoding, CPPNs (Stanley, 2007), was used to generate soft voxel-formed three-dimensional structures (see Fig. 3.5), coupled with the use of NEAT algorithm which ensures the increasing complexity of the networks produced. The superiority of this kind of generative encoding was verified, showing how CPPNs can take advantage of their geometrical properties. Evaluation was done by a simple

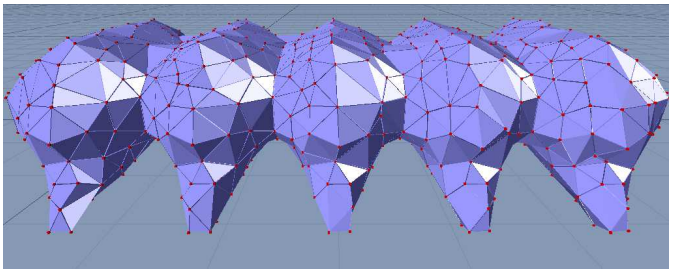


FIGURE 3.6: Soft robot bodies are built out of meshes of tetrahedra (Rieffel et al., 2014).

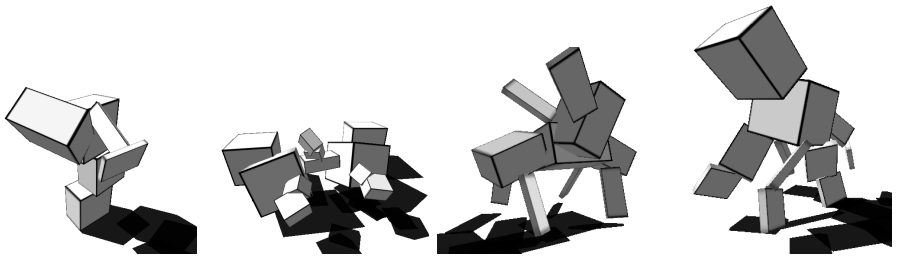


FIGURE 3.7: Diverse morphologies evolved during a single run of novelty search with local competition (Lehman and Stanley, 2011b).

displacement measure, while evolution tended to evolve different kinds of locomotion strategies and morphologies as the fitness function was penalized for different kinds of parameters. Furthermore, it has been shown that evolving morphologies (CPPNs) in lower resolutions and then applying the same networks for higher resolution structures can be beneficial, since the locomotion behaviors in lowers structures also apply in higher saving computational time. An earlier work (Hiller and Lipson, 2010), apart from the generative encoding of CPPNs, made use of *Gaussian Mixture* and *Discrete Cosine Transform* to produce amorphous soft body structures.

The simultaneous evolution of soft robot morphology and control was also investigated by recent work (Rieffel et al., 2014) (see Fig. 3.6). Some aspects of soft robot evolution were verified in this work, namely muscle placement and muscle-firing patterns can be evolved given a fixed body shape and fixed material properties. Furthermore, material properties can be co-evolved alongside locomotion strategies. Finally, a developmental encoding was introduced, allowing more complex parts to be added to soft robotic structures during the evolution.

3.2 Evolving Virtual Creatures by Novelty Search

In problems with such high dimensionality as evolving both the morphology and locomotion strategy of artificial creatures in simulated or physical environments, evolution does not explore the solution space enough sticking with the first most promising morphologies to exploit. However, novelty search, a technique that explicitly rewards diversity, can potentially mitigate such convergence. Methods for evolving such virtual creatures like in (Sims, 1994) can utilize novelty search (Lehman and Stanley, 2011b) and be far more explorative in the search space (see Fig. 3.7). Behavior novelty defined as a measure between morphological properties of the produced creatures driving the evolution to explore more diverse morphologies. A larger diversity with regards to the morphological properties of the evolved virtual creatures does not guarantee their ability to locomote in

the simulated environment. However, combining fitness and novelty objectives through local competition led to improved results, whereas novelty search alone failed.

As it was stated before, the work that is being presented in this thesis makes use of novelty search to co-evolve the morphology and the locomotion capability of soft bodied virtual creatures. As pure novelty search failed to evolve fit solutions in previous work (Lehman and Stanley, 2011b) used, it is of interest to apply and investigate its performance in virtual soft robots this time.

Chapter 4

Method

In this chapter an introduction to the problem specifications and a comprehensive documentation describing the methods used is given. As an initial experiment, a random methodology to generate soft robot morphologies is implemented to verify that random non-evolutionary approaches fail in such settings. Next, evolutionary methods are used in order the morphology and the locomotion strategy of soft robots to be simultaneously evolved in the simulated environment. A direct encoded genome is used within a simple genetic algorithm as in (Cheney et al., 2013). This direct encoded GA is expected to be unable to evolve efficient locomotion due to the irregular morphologies direct encoding is producing. Furthermore, a generative encoding method is used and paired with the NEAT evolutionary algorithm in order to establish a baseline for the following experiments, verifying previous work (Cheney et al., 2013).

Pure novelty search was applied in the evolution of three-dimensional virtual creatures in a simulated environment (Lehman and Stanley, 2011b) using as a behavior metric the morphology of the produced creatures, where it failed to compete with the traditional fitness based search method. Novelty search is the main search methodology investigated in this thesis. The implementation of this method alters the pipeline of the NEAT and any evolutionary algorithm to fit the new objective function. In addition to the discussion regarding the algorithm of novelty search, different behavior metrics that can be used in this problem task are defined.

4.1 Problem Introduction

Recent work in evolutionary robotics shows that compositional pattern-producing networks (CPPNs) can encode soft robot morphologies. These networks can produce regular

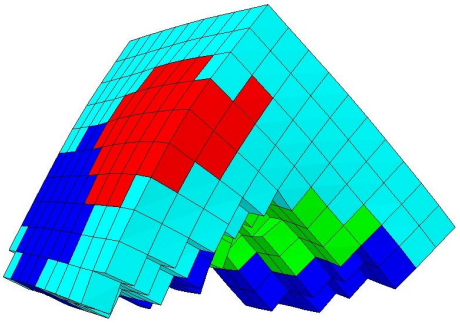


FIGURE 4.1: Soft robot uses four materials (two active, two passive), morphology evolved penalizing actuated materials.

outputs which are translated to regular shapes for the structures produced, resulting in efficient locomotion for the evolved virtual soft robots. VoxCad simulator provides a test-bench for analyzing soft robot bodies that can be actuated through environmental changes, which is the temperature variation in the specific setting. In addition to that, recent work by (Cheney et al., 2013) showed that very interesting morphologies can be evolved by the CPPN-NEAT algorithm in the specific soft robot simulation environment.

VoxCad

For the simulation of the soft material bodies, VoxCad's (Hiller and Lipson, 2012a) underlying physics engine *Voxelyze* was used as a stand-alone software to analyze the soft structures without the computational cost of rendering. As far as the soft material simulation settings are concerned, this thesis is not aiming at finding the best environmental and material properties. All variables of the environment excluding the temperature period and the gravity acceleration are constants throughout this thesis. Table A.1 describes and presents the values used in different variables of the simulation.

Materials

Within the VoxCad simulation software there is the option of defining and using a palette of materials. Materials can be *passive* or *active*. Passive materials do not react to temperature changes, while active materials expand and contract in respect to their thermal properties. Figure 4.1 illustrates a soft robot consisting of all four materials are used in the experiments. Red and Green are the only actuated materials with non-zero and opposite thermal expansion coefficients, meaning that their phase in respect to the actuation from temperature changes is equal to half a circle. Green voxels contract the

same time red expand and vice versa, mimicking living organisms' muscle tissue. The two additional materials represent soft non-actuated tissue that can be soft (soft tissue) or hard (bones). Cyan voxels are soft having five times smaller elastic modulus of their material than Blue which have 50 `MPa`.

4.2 Random Generation of Soft Robots

To evaluate all the following evolutionary methods used, information about the performance of random generated morphologies must be present. In order to achieve that, two random approaches which will also help the understanding between direct and indirect encoding are implemented.

The first implementation of a random morphology generator mimics "direct" encoding. This method assigns randomly the presence of a voxel in the three-dimensional space of the lattice. The probability of adding a voxel in given coordinates is 0.5. The material of each newly added voxel is chosen randomly from the material palette. Each material has the same probability of being chosen. After all voxels have been added and assigned a material, unconnected parts of of the structure will be removed keeping only the largest connected structure in the lattice.

An "indirect" way of generating random morphologies follows a different method of assigning materials to voxels, adopting a set of rules in order to generate a new soft robot morphology. This method holds two probabilities, the one refers to the probability of adding a new voxel in the already generated structure, the next one denotes the probability that the material of a new inserted voxel will be the same as the one of the material is going to be connected to. First, a random material voxel is inserted in a random coordinate into the lattice space. When a new voxel is to be added, a connection (voxel) is chosen from the already added voxels. The side of the connection is chosen from a uniform distribution out of all valid (within the lattice space) sides. In this generative process there is also the possibility of creating structures in half of the lattice space and then mirror the soft structures in both halves of it, generating in this way symmetrical morphologies.

Considering these three methods the difference between direct and indirect coding presented in Section 2.3 is becoming easier interpreted. In the "direct" process a probability determines the presence and the material for every coordinate in the lattice space. On the other hand, the "generative" method holds a set of rules and probabilities defining the structure that is going to be produced in the available space.

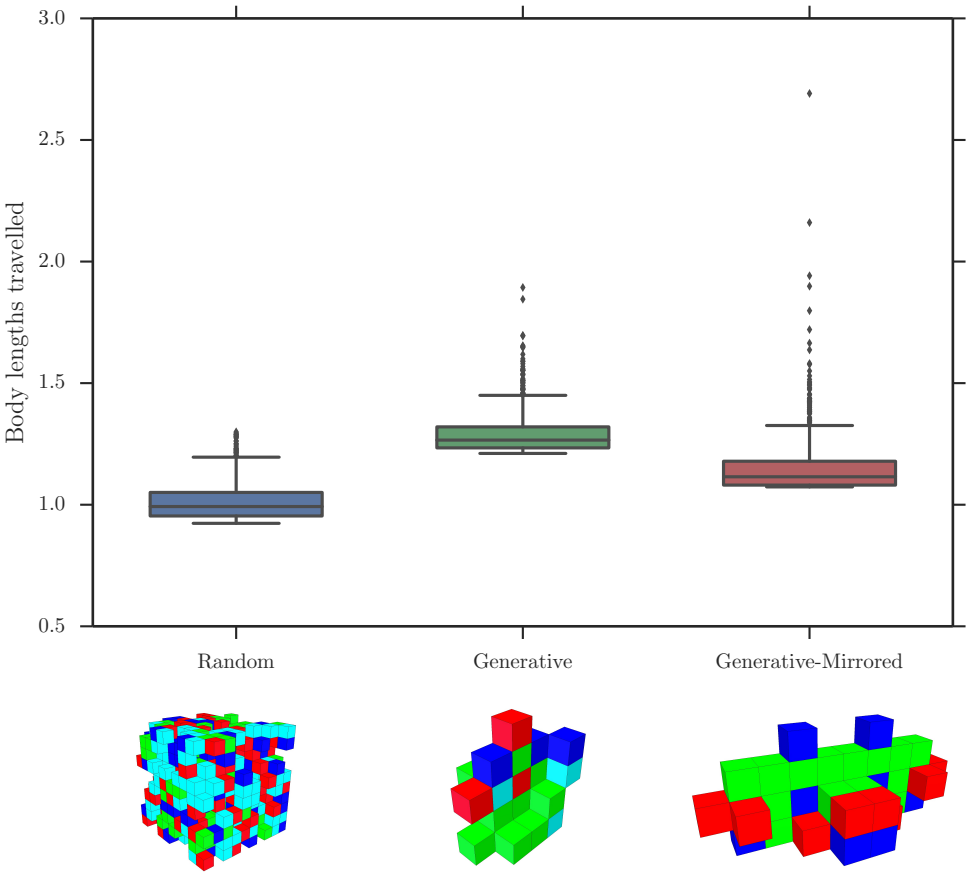


FIGURE 4.2: Generative encoding creates more natural morphologies even in random schemes. (see Settings B.3)

Figure 4.2 illustrates not only the actual performance (fitness in body lengths traveled by top-1000 soft robots from 30000 total runs for each method) of the previously described methods, but also one of the best performing soft robots of each method. Both "generative" methods outperform the "direct" one due to the fact that they are capable of generating regular morphologies. "Generative" random soft robot generation methods create more compact structures which can move easier due to their size and their geometrical features. For the *Generative-Mirrored* approach even though the average performance is slightly worse than the plain method it actually performs way better in some distinct cases (i.e outliers). Adding geometrical properties resulted in getting more efficient locomotion by the generated soft robots.

All methods achieved an average displacement of the soft robots around or more than one body length, which is considered to be very low in respect to the robots generated by evolutionary methods are presented later in this thesis.

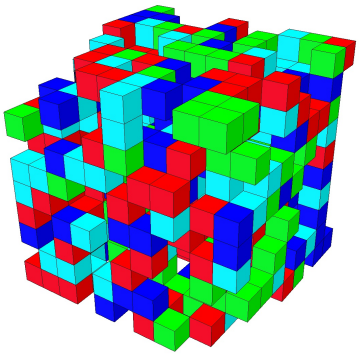


FIGURE 4.3: Direct encoding cannot capture the geometrical properties of some problems.

4.3 Direct-Encoded Evolutionary Soft Robots

In the previous section three random methodologies of "indirect" and "direct" generation of soft robots were implemented, failing to produce any decent locomotion gaits for the soft structures. Considering how vast the solution space is, random approaches are doomed to fail in a definite number of tries. Therefore, a more sophisticated evolutionary method is discussed here.

Direct encoded genomes coupled with a simple genetic algorithm is a successful approach in evolving robot controllers. As it was previously stated in Chapter 2, mutations and crossovers of real-value streams search the problem space effectively finding near optimal solutions in demanding optimization problem domains. The GAlib C++ library (Wall, 1996) is used for the implementation of this method.

Representation of the genotype

As in every direct encoding scheme genotype is represented by a stream of bits, which length is equal to the number of dimensions of the problem. The number of the materials in the palette are the first dimensions of the problem. The presence or not of a voxel at a given position in the lattice space adds one more dimension to the representation. Analytically, its length can be represented by a stream of length equal to the number of voxels in the lattice times the materials used plus one denoting the presence of the voxel, the length of the genome is described by the following equation:

$$|Genome| = (l_x \times l_y \times l_z) \times (1 + |p|) \tag{4.1}$$

where l_x, l_y, l_z are the dimensions of the lattice space, and $|p|$ is the size of the palette of materials.

$$Genome = \underbrace{01010\ldots011011}_{\text{Presence}} \quad \underbrace{10101\ldots110011}_{Material_1} \quad \ldots \quad \underbrace{00011\ldots111110}_{Material_n}$$

The above stream of bits illustrates how a soft structure in VoxCad environment can be represented by a direct encoding scheme. Each of the values of the stream is represented by a float value between zero and one, which is represented by a stream of bits in lower-level. The mapping from the genotype level to the phenotype is straightforward in this case, the first stream of values is used to determine the presence of a voxel in given coordinates while in case of presence the other n streams are used and the maximum value in specific positions of the streams determine the material is going to be used.

Considering the representation of the genome, as well as the geometrical nature of the problem itself it is not expected that direct encoding will capture this major property of the problem (see Fig. 4.3). Therefore, it is anticipated that direct encoded genomes will not be able to generate soft robots that can produce efficient locomotion in these settings (Cheney et al., 2013).

4.4 Generative-Encoded Evolutionary Soft Robots

Direct encoding methods lack the morphology regularities of the soft robots evolved by this method. Compositional pattern-producing networks can serve this function. CPPNs are built up by a set of canonical functions which enable the outputs of the network to produce repetitive, symmetrical and geometrically interesting patterns. Producing regularities in the phenotype space and capturing geometrical properties of the optimization problem, it is expected that this representation is going to produce efficient locomotion strategies and morphologies of the soft structures (Cheney et al., 2013). Since, CPPNs must be queried for every coordinate of the lattice space, the input nodes (neurons) of the CPPN are assigned to x,y,z normalized coordinates following (Cheney et al., 2013), so that:

$$x, y, z \in [-1, 1]$$

A bias input node is also introduced in the genome CPPN representation, this will allow the network to produce arbitrary outputs different from the defaults when all other inputs values are set to zero. More inputs could be added to the CPPNs, for instance the distance from the center point of the Cartesian phenotype space (lattice) as described in (Stanley, 2007) and used in (Cheney et al., 2013), which naturally adds more bias towards symmetrical structures. However, the evolution of such aesthetic structures is

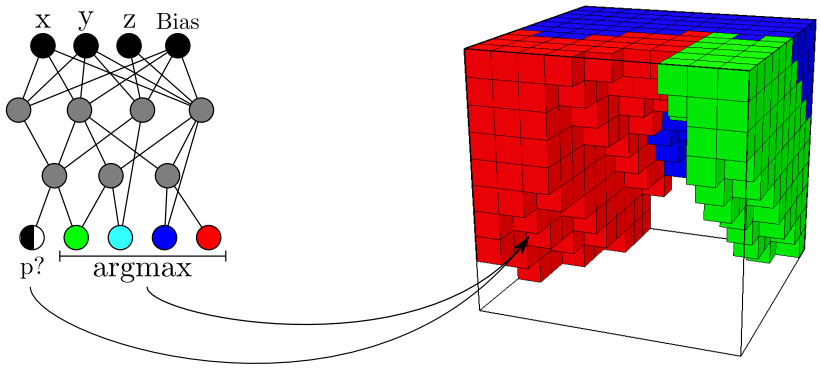


FIGURE 4.4: Each genotype (CPPN) is queried for every coordinate inside the lattice, its outputs determine the presence of a voxel and the type of its material.

not much of interest to exploit. CPPNs, as it is shown later in this thesis, can evolve symmetrical morphologies without this extra information input node(s). The proposed input nodes for the three dimensions of the Cartesian space provide the minimum bias to the network outputs. Figure 4.4 illustrates the topology of a random CPPN network with the input and output nodes previously described. The set of nodes and connections determine the *topology* of the network. The topology of these networks can be variant and be evolved alongside the weights of the connections in any neuroevolution method. The divergent part of the network between the input and the output nodes is described by the genotype and it is the one that is going to be evolved and altered during the evolution. The presence of a voxel in each coordinate of the lattice is determined by a single output of the CPPN, denoted with p while the selection of the material is determined by n-outputs. The node with the maximum value out of the n-outputs will determine which of the materials is going to be used in the specific voxel only in cases this is present.

4.4.1 Evolution of Generative Encoded Genomes

The evolution of these indirect representations of the genotypes can be evolved with any method able to evolve artificial neural networks, since these are identical to CPPNs. CPPN-NEAT (see Sec. 2.3.1), is a method to evolve CPPNs with the NEAT evolutionary method. Previous work (Cheney et al., 2013), showed that this method can indeed evolve the morphologies of the soft robots in the VoxCad simulation environment. *HyperNEAT*[1] is used for the implementation of the CPPN-NEAT algorithm. Algorithm 4.1, presents the pseudocode for the evolution under CPPN-NEAT method. In addition, a brief explanation of the function used in the algorithm follows:

[1]HyperNEAT `v4.0 C++` by J. Gauci code (url: `https://github.com/MisterTea/HyperNEAT`)

Algorithm 4.1 CPPN-NEAT evolution

```
1: population = ∅
2: species = ∅
3: generation[0] = initial_population()
4: for i = 0 to max_generation do
5:    species = species ∪ speciation(generation[i])
6:    evaluation(generation[i])
7:    adjust_fitness(generation[i], species)
8:    selection(generation[i], species)
9:    generation[i + 1] = reproduction(generation[i])
10:   population = population ∪ generation[i + 1]
11: end for
```

Initial population Before the evolution starts, an initial population must be produced, identical genomes (CPPNs), with variant connection weights fill up the population.

Speciation Takes place and split the population in separate species or adds individuals to already existing species in respect to their networks' topologies; a compatibility function determines the similarity between two genomes (see Eq. 2.1). However, all firstly introduced genomes belong to the same species, due to the identical topology of their CPPNs.

Evaluation Once the population is filled with new individuals, these have to be evaluated. Simulation is taking place for each of the individuals of the population, where each one of them is awarded with a fitness value.

Fitness adjustment After all individuals are evaluated, each species is assigned a value which is the sum of the fitness values of the individuals belonging to this species divided by the number of the individuals. This way, it is been decided how many individuals each of the species will breed and it is directly determined by the average fitness of each species.

Selection As soon as the number of new individuals each species is determined, only the top 20% of the species population will reproduce, the rest population will "die". *Competition*, and *Elitism* as other genetic selection techniques can also be used in this step of the evolution.

Reproduction There are two ways for the selected individual inside each species to reproduce. These are *mutation*, which slightly changes the genome of one parent to create a new genome, and *crossover*, where two parents combine their genes to create a new individual.

Algorithm 4.2 CPPN-NEAT with novelty search

```
1: population = ∅
2: novel_inds = ∅
3: species = ∅
4: generation[0] = initial_population()
5: for i = 0 to max_generation do
6:   species = species ∪ speciation(generation[i])
7:   evaluation(generation[i])
8:   for all ind ∈ generation[i] do
9:     novelty = sparsity(ind, (generation[i] − ind) ∪ novel_inds)
10:    if (novelty ≥ novelty_threshold || novel_inds == ∅) then
11:      novel_inds = novel_inds ∪ ind
12:    end if
13:  end for
14:  adjust_novelty(generation[i], species)
15:  selection(generation[i], species)
16:  generation[i + 1] = reproduction(generation[i])
17:  population = population ∪ generation[i + 1]
18: end for
```

4.4.2 Novelty Search

Novelty search, as first presented in Chapter 2, requires only small changes in the pipeline of an evolutionary algorithm. Fitness is replaced by a novelty metric which determines how novel is a phenotype's observed behavior with respect to all novel behaviors found earlier in the evolution. Sparsity (see Eq. 2.2) is used to determine this value while every individual is compared not only with the previous novel behaviors, but also with the observed behaviors by individuals from the current generation. The algorithmic adjustments within CPPN-NEAT algorithm are indicated in Algorithm 4.2 (Red colored text), where the pseudocode of novelty search is presented.

Evaluation function in fitness based evolution was responsible of evaluating an individual in respect to an objective. This objective function is equal to the body-lengths traveled by the soft robot within a specific time span. However, the same function is now also responsible for observing the behavior of each individual and record it. In this way, the novelty of a behavior can be computed based on recorded behaviors of other individuals. Function **sparsity** computes the sparseness (see Eq. 2.2) of a specific individual's observed behavior in the behavior space. Following the evaluation of each individual, its behavior will be compared to all novel behaviors stored during the evolution. A threshold determines if the observed behavior is considered novel in respect to the set of behaviors was compared to. The fitness adjustment of the previous code example is becoming **novelty adjustment** following the same functionality, selection and reproduction operations can be applied in the same way.

4.4.2.1 Behavior in novelty search

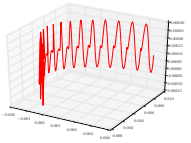
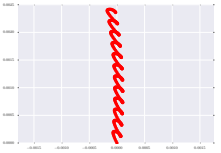
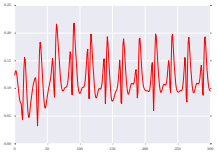
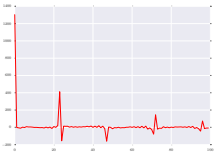
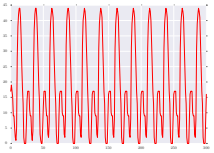
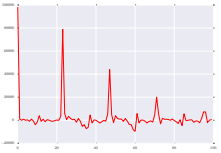
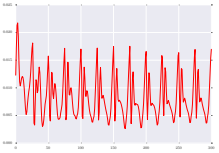
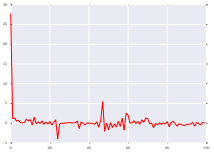
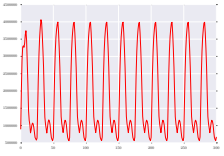
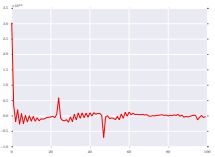
Behavior can be defined as the way that a human/machine behaves towards or within an environment. Regarding the evolution of soft robots in the specific simulated environment, a behavior can be defined as the way soft robots behave in respect to their locomotion strategy. Every aspect of the soft robots movement that can be observed can be used to describe their behavior. Previous work (Lehman and Stanley, 2011b) in a try to evolve walking three-dimensional virtual creatures used the evolved morphology of the creatures to describe their behavior. Although, in this work comparing the morphology of the evolved soft robots is similar to comparing the chromosome (CPPN) of each individual. Therefore, only the comparison of the observed behavior in phenotype level can lead the evolution towards more complex behaviors.

A straightforward function that determines the goodness of an individual is used in fitness based methods. This measure drives the search in "good" areas of the search space. However, the same measure cannot be used for novelty search. What novelty search looks for is novelty in behavior space. It is expected that behaviors that contain information about the goodness (displacement) of individuals will be more successful than behaviors that include other aspects of the soft robots' behavior. In cases that behavior does not contain information about the objective, the search for novelty will become random in regards to this objective function.

Behaviors that describe the morphology of the evolved robots have failed (Lehman and Stanley, 2011b), since search is then forcing new types of morphologies without caring about the actual target of the evolution, which was the efficient locomotion. To present a similar idea consider a behavior metric that enumerates the number of voxels a soft robot consists of. This is not a well-founded behavior metric, since the search will reward new structures with different number of voxels from previous evolved structures. Therefore, there will not be exploration in the behavior aspect that affects the actual target of the evolution, which is to produce and evolve efficient locomotion strategies.

Table 4.1 presents all behaviors used for the novelty metric computation together with the sampling rate of the recorded values during the simulation and their description. For all recorder behavior metrics a constant sampling rate ensures that all signals have the same length. The behaviors designed to describe the strategy and the efficiency of the evolved locomotion. They contain information that indirectly implies both the objectives of the evolution. *Trajectories* (2D and 3D), incorporate all the needed information such as speed, displacement, and locomotion strategy. To avert from same trajectories in all possible directions trajectories are normalized, meaning that their starting coordinates are always the start of the axes ($< 0, 0, 0 >$) and the point coordinates of the trajectory

TABLE 4.1: Observed behaviors of the soft robots used for the sparsity computation in novelty search.

Behavior	**Sampling**	**DFT**	**Example**	**Description**
3D-trajectory	1 KHz			Set of three-dimensional sampled points of the robot's center of mass during simulation.
2D-trajectory	1 KHz			Set of two-dimensional ground projection sampled points of the robot's center of mass during simulation.
Pace	1 KHz			Set of robot's pace sampled values.
DFT-Pace	100 KHz	✓		Set of the robot's pace sampled values transformed into the frequency space.
VTG	1 KHz			Set of voxels touching the ground in each sampling time.
DFT-VTG	100 KHz	✓		Set of voxels touching the ground transformed into the frequency space.
Pressure	1 KHz			Set of maximum pressure among the connected voxels.
DFT-Pressure	100 KHz	✓		Set of maximum pressure among the connections transformed into the frequency space.
KE	1 KHz			Set of maximum kinetic energy of voxels.
DFT-KE	100 KHz	✓		Set of maximum kinetic energy of voxels transformed into the frequency space.

are rotated so their center of mass is normalized to a specific angle ($\theta = 90^o$). To measure the difference of two trajectories the Euclidean distances between coordinates at the same sampling time are measured, so that:

$$\text{First trajectory: } t_i = t_i^1, t_i^2, \ldots, t_i^N \tag{4.2}$$

$$\text{Second trajectory: } t_j = t_j^1, t_j^2, \ldots, t_j^N \tag{4.3}$$

$$\text{Difference: } t_i - t_j = \sum_{n=1}^{N} dist(t_i^n, t_j^n) \tag{4.4}$$

where n is the number of sampled coordinate points and *dist* is the Euclidean distance. *Pace* is also a very informative behavior metric as it directly measures the speed of the robot. *Voxels touching the ground* can also imply information about the locomotion strategy but not enough about the actual performance regarding the displacement. Hopping robots that move fast can have same behaviors with regards to this metric with hopping robots with zero speed. *Maximum pressure* among the voxels' connection is yet another behavior metric, pressure is expected to become higher as structures move faster and interactions with the ground eventually getting harder. Finally, *maximum kinetic energy* is a behavior metric that straightly determines the displacement of the voxels in the structure. To compute the difference between two signals, a straightforward method is used. Subtracting the one signal from the other, taking the absolute differences, and summing them up to compute one single value that describes how variant the two signals are. More specifically:

$$\text{First 1-d signal: } s_i = s_i^1, s_i^2, \ldots, s_i^N \tag{4.5}$$

$$\text{Second 1-d signal: } s_j = s_j^1, s_j^2, \ldots, s_j^N \tag{4.6}$$

$$\text{Difference: } s_i - s_j = \sum_{n=1}^{N} |s_i^n - s_j^n| \tag{4.7}$$

For all behaviors but trajectories, the Fourier profile of their signals can also be used as an observed behavior. This process of transformation of the one-dimensional signals into frequency space eliminates shifts of signals in time-axis. The discrete Fourier transformation:

$$C_i^k = \sum_{n=1}^{N} s_i^n e^{-i2\pi kn/N,}, \ \forall k \in \mathbb{Z} \tag{4.8}$$

For the Fourier transformations of these signals the first twenty coefficients are compared, and the summation of their absolute differences determines the difference of the two behaviors.

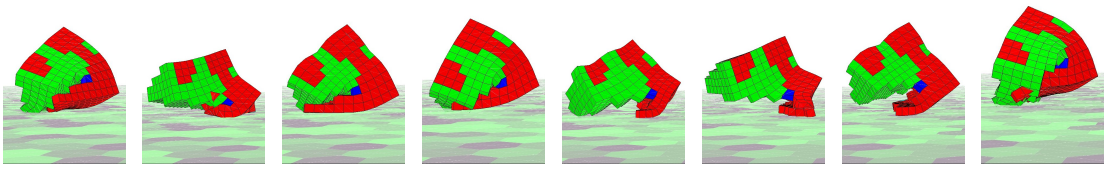
$$\text{Difference in frequency: } s_i - s_j = \sum_{k=0}^{R-1} |C_i^k - C_j^k|, R = 20 \tag{4.9}$$

In this section, different observed behavior metrics have been defined. The similarity or difference of two same type behaviors can be determined by the equations provided while these measures of difference are used by the sparsity equation (see Eq. 2.2) to compute the sparseness of a given behavior in the behavior space. Individuals with novel observed behaviors (high sparseness value) are then stored in a list helping the evolution to avoid generating similar behaviors (see Alg. 4.2).

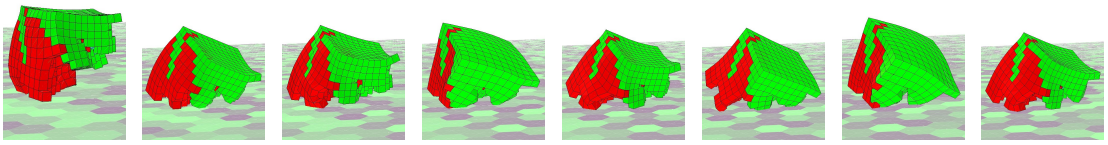
Chapter 5

Results & Discussion

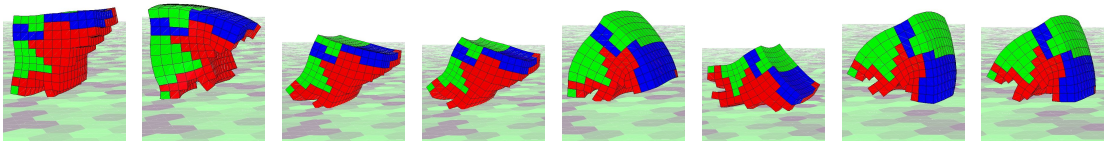
This chapter presents the performance and the resulted evolved virtual soft robots by the methods described earlier in this thesis (see Ch. 4). In addition, the performance and significant findings of the evolutionary methods used for the co-evolution of the morphology and the locomotion strategy of soft robots are discussed in details. In the previous chapter it has been shown how random generated soft robot morphologies fail to produce any locomotion capabilities in this setting. Thus, the results obtained by these random methods are not discussed in this chapter. The performance of evolutionary methods is only presented here. Pure novelty search is compared in respect to the goodness measure used in the simulations (displacement of soft robots in body-lengths) to fitness based search. The effect that novelty search has in the average and champion fitness of the population during the evolution is investigated in detail in the following sections. Additionally, both search methods are compared with respect to the number of novel behaviors they evolve during an evolution run. Moreover, the influence of the behavior metric in novelty search is shown, where all behavior metric proposed earlier are used to define the novelty of an individual. An altered number of closest behaviors in the sparsity equation of the novelty search, leads to an interesting conclusion about the effect it has in the evolution process. Genetic selection techniques such as competition and elitism are also used to improve the baseline methods. More specifically, elitism is used in a proposed methodology to incorporate fitness information in novelty search. Last, the performance of both methods are investigated for several levels of gravity. This will show that gravity conditions do not have any effect in favor of a specific search method. Furthermore, evolved locomotion strategies under different gravity conditions show how environmental conditions can affect the evolved morphologies and the strategies of soft robots.



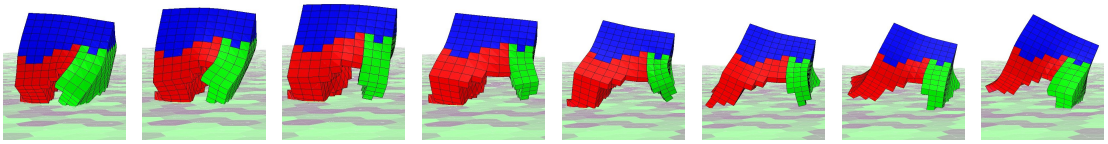
(A) 2-legged pull locomotion



(B) 4-legged (nose & tail) animal-like locomotion



(C) 2-legged push-pull locomotion



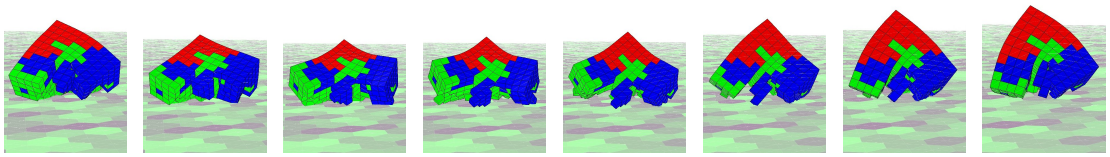
(D) 2-legged galloping

FIGURE 5.1: Champion (best overall) morphologies evolved in independent runs of fitness based search. Each row illustrates the locomotion strategy of the individuals created. (Settings B.3)

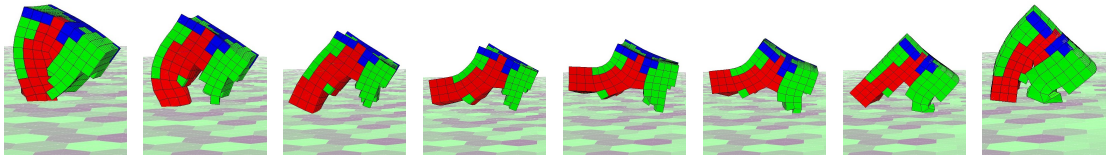
As in (Cheney et al., 2013) and for comparison purposes a population of 30 on each generation is used, the maximum number of generations in the evolution is set to 1000. For more details about the setting are used in the evolutionary algorithms, see Appendix C. For simulation settings used see Appendix A. Due to computationally expensive simulations, not all experiments are performed using a lattice resolution of 10^3, resolutions lower than 10^3 are used as well. More specifically, $5^3, 7^3, 10^3$ lattice resolutions are used, see Appendix B.

5.1 Evolved Morphologies

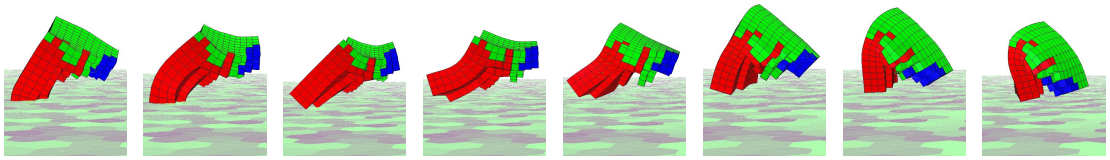
In this section some of the evolved morphologies and their effective locomotion patterns evolved within fitness based and novelty search will be discussed. Apart from the performance that the two methods achieved, both of them were successful in evolving effective



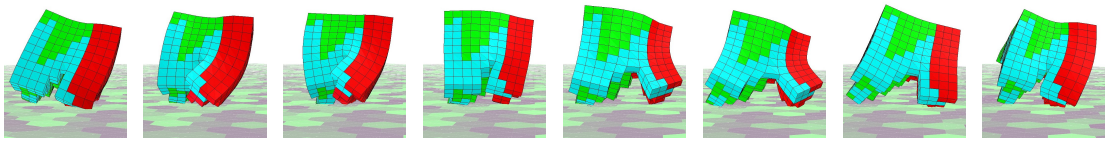
(A) 4-legged animal-like locomotion



(B) 2-legged galloping



(C) L-shaped hopper



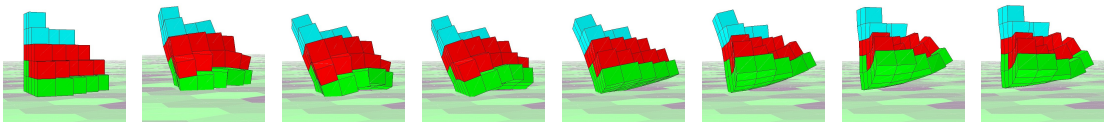
(D) 2-legged galloping

FIGURE 5.2: Champion morphologies evolved in independent runs of novelty search. Each row illustrates the locomotion strategy of the individuals created. (Settings B.3)

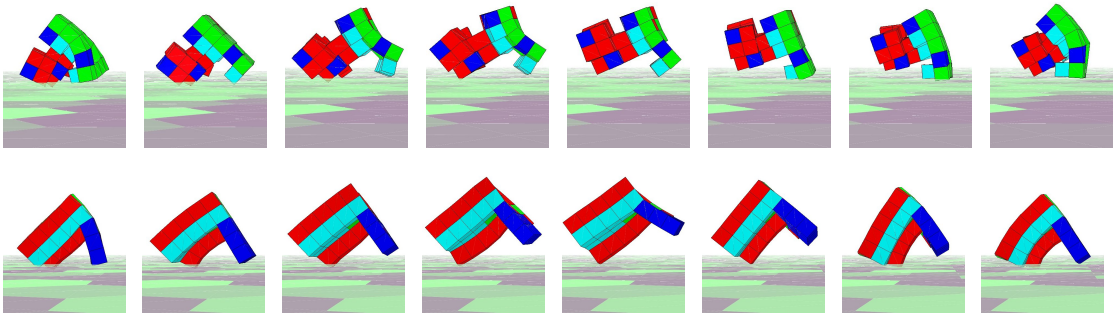
strategies for the locomotion of the evolved morphologies. Figure 5.1 shows four different gait types evolved by fitness based search. All of these morphologies are considered "good" in respect to their fitness value, meaning that they achieve to travel up to ~ 10 body lengths during the simulation time (0.4 sec.). Considering that the lattice space used for this experiment was of size 10^3. The produced low-resolution soft robots cannot be compared with real-life organisms. However, the results are shown that even in such low dimensions life-like locomotion can be evolved.

For the fitness based search, soft body morphologies can use their front leg(s) to pull themselves forward (see Fig. 5.1a), evolve a four-leg locomotion where a nose and a tail are mostly used for stability (see Fig. 5.1b), push and pull themselves forward (see Fig. 5.1c), and gallop using both of their legs (see Fig. 5.1d).

Moving from fitness based search to novelty search locomotion strategies do not differ too much since the resolution does not allow the virtual soft robots to explore more locomotion techniques. However, novelty search proves its merits with regards to the



(A) Fitness based search



(B) Novelty search

FIGURE 5.3: Champion morphologies evolved in independent runs of fitness based and novelty search in a lower resolution (5^3). Each row illustrates the locomotion strategy of the individuals created. (Settings B.1)

morphologies evolved. More complicated structures are now evolved, this can be explained by the fact that novelty search pushes the evolution to investigate new kinds of behaviors resulting to more complex topologies for the networks (CPPNs) that represent the soft robot morphologies. Figure 5.2 presents four champion morphologies and their locomotion strategies. Once again, two-legged galloping soft robots (see Figs. 5.2b, 5.2d), animal-like locomotion based on four legs (see Fig. 5.2a), and hopper soft robots (see Fig. 5.2c) are evolved.

Having presented the types of locomotion patterns have been evolved in the specific resolution for the lattice (10^3), it is of interest to see what both search techniques can achieve in a lower resolution setting. Figure 5.3 illustrates the results for both methods in a lower resolution setting (5^3). Both methods achieve in evolving *fit* soft robots which can locomote efficiently. What is interesting though, it is the fact that all experiments held by fitness based search failed to produce the locomotion strategies evolved by novelty search. A "quarter-pyramid" shaped soft robot (see Fig. 5.3a) was the champion individual in almost all runs of fitness based evolution in this setting, whereas novelty search came up with two-legged virtual creatures (see Fig. 5.3b).

Different locomotion strategies have been evolved under two different methods, using the same settings. The discussion following in the next sections is mostly focused on the performance comparison of novelty search against the traditional fitness based method within CPPN-NEAT evolutionary method.

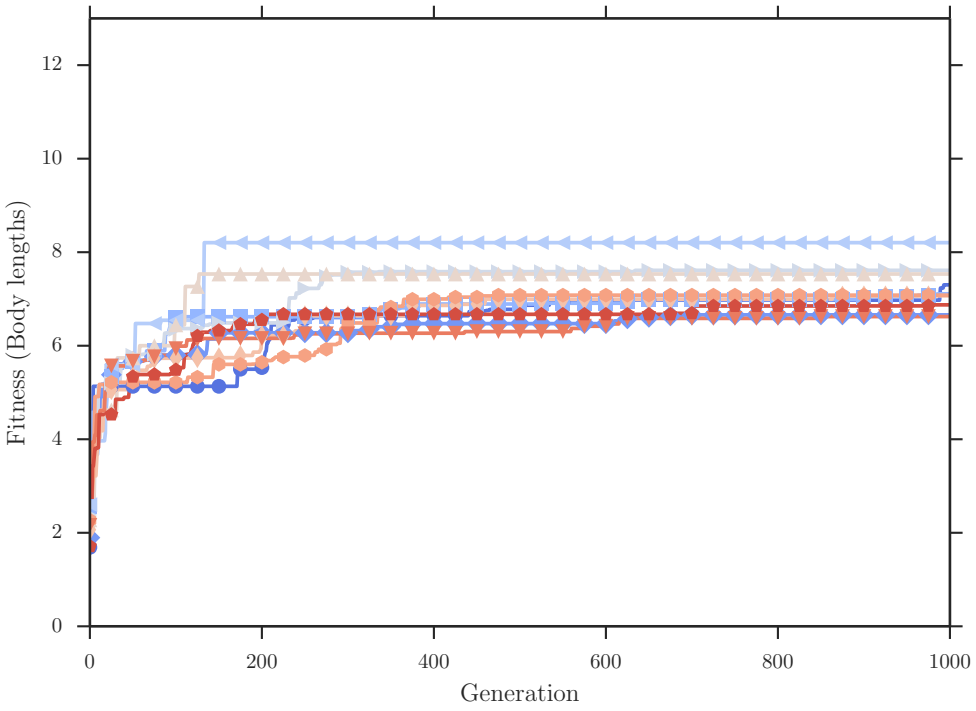


FIGURE 5.4: Best fitness so far, 10 individual runs for fitness based search. Each line is a different run. (Settings B.2)

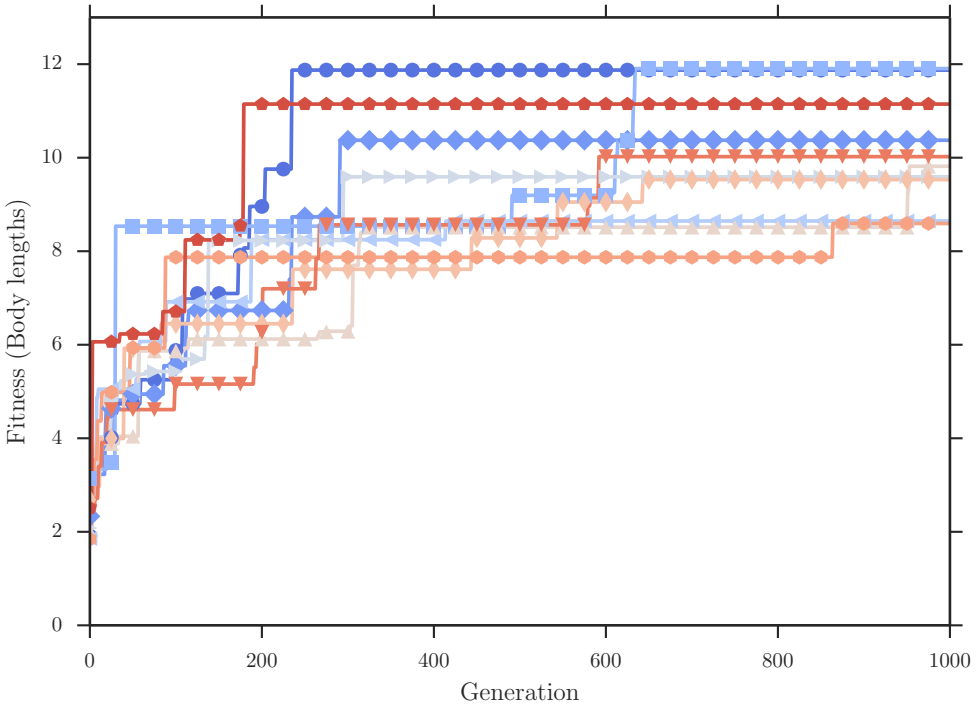


FIGURE 5.5: Best fitness so far, 10 individual runs for novelty search. Each line is a different run. (Settings B.2)

Table 5.1: Evolutionary methods

Method	**Encoding**	**EA**	**Selection based on**
Novelty Search	*CPPN* (Indirect)	*NEAT*	Sparsity
Fitness Search	*CPPN* (Indirect)	*NEAT*	Displacement in body lengths
Random Search	*CPPN* (Indirect)	*NEAT*	Random
Fitness Search - D.E.	`bit`-stream (Direct)	*GA*	Displacement in body lengths

5.2 Into The Performance of Novelty Search

Before comparing novelty search to fitness based search, it is of interest to show how they individually behave under the same simulation settings. Figure 5.4 shows 10 independent runs for fitness based search. Following the gradient of the objective function fitness based evolution does small steps towards better and more optimized solutions from generation to generation. However, fitness based evolution often focuses on specific morphologies which then tries to optimize leading the evolution to stop at these local maxima.

Figure 5.5 shows 10 independent runs for the novelty search under the same settings. When compared to the fitness based search (see Fig. 5.4) a clear difference can be observed. Evolving for novelty means that within the evolution novel behaviors are rewarded instead of fit behaviors or behaviors that lead to the optimization of an objective function. Fit individuals in respect to the objective function, for which novelty search has no information within the evolution process, are results of new novel behaviors that novelty search looks for. Observing only big steps in the fitness, it is valid to say that there is no optimization of specific morphologies within novelty search. Initially, novel individuals are highly rewarded, these individuals can be very good in respect to the fitness or not. In the next generations, mutations, crossovers, and copies of these novel individuals are not going to be highly variant in respect to their chromosome from their ancestors, resulting to similar behaviors. These comparable behaviors are not going to be remarkably rewarded in respect to their novelty value. Thus, highly novel individuals are producing less novel children in regards to their behavior. These children, while their fitness can be higher than their ancestors' and still having the potential to be optimized further, will not have the chance to reproduce in the next generations and be further improved in regards to the objective function.

To extensively compare the performance achieved by novelty search method, its performance is set side by side with fitness based search, random search, and finally a simple genetic algorithm with direct encoded genomes. The same experiment held under two different simulation settings (for resolutions 5^3 and 10^3). Notice, that the first three methods are referring to a generative encoding (CPPNs) evolved by CPPN-NEAT

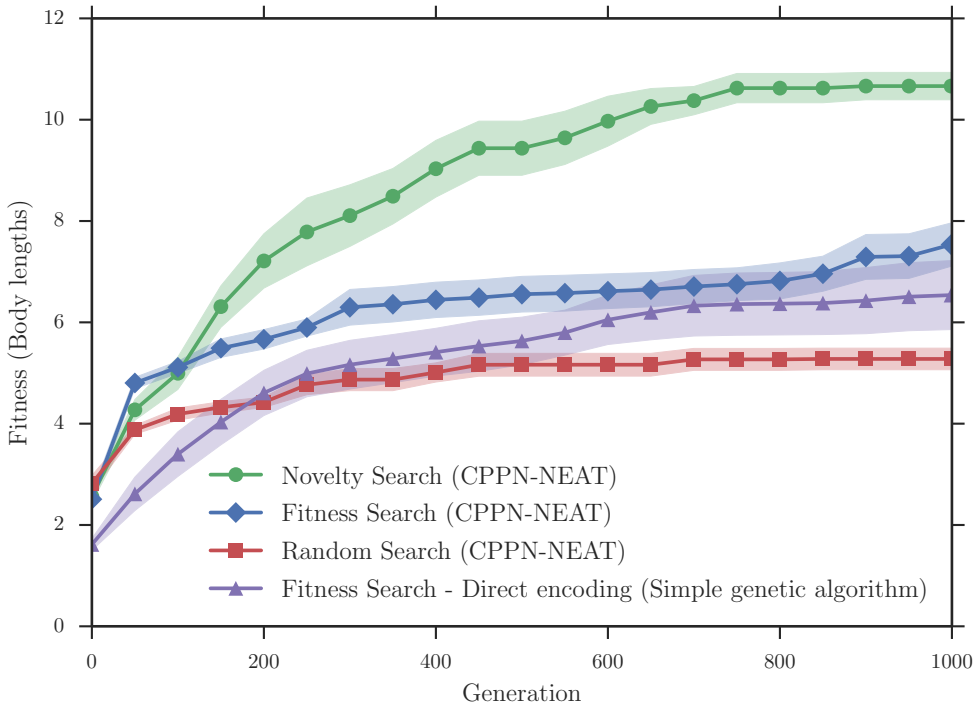


FIGURE 5.6: Comparison of simple genetic algorithm (direct encoding) against novelty-fitness-random search with generative encoding. Best fitness so far averaged over 10 runs. (Settings B.1)

evolutionary algorithm and using selection in respect to novelty, fitness, and random selection. The last method uses a direct encoded genome driven by fitness within a simple genetic algorithm. Table 5.1 presents the methods used for the evolution of soft robots. Two-dimensional trajectories, as described in the previous chapter (see Sec. 4.4.2.1), are used by novelty search in order to describe the novelty in the behavior space through sparsity equation. The objective function that describes the goodness of solutions is the displacement of the soft robot's center of mass from its initial position in body-lengths and it is used for all fitness based methods. Random selection in CPPN-NEAT achieved choosing random selected individuals to breed on each generation. For direct encoding, direct encoded genomes represent the solutions as described in Section 4.3.

Figure 5.6 presents the results for the low resolution soft robots (5^3). The average best displacement so far of the soft robots in body lengths is presented alongside the deviation error. Notice, the difference between novelty search and the other methods. Novelty search evolves structures that are superior than any other method does in these settings. It should be mentioned that in such a small structures complex locomotion patterns cannot be evolved due to stability issues of the simulator and because of the fact that lightweight structures can be bouncy leading to ball shaped structures capable of achieving large displacement from their initial positions. That being said, we still

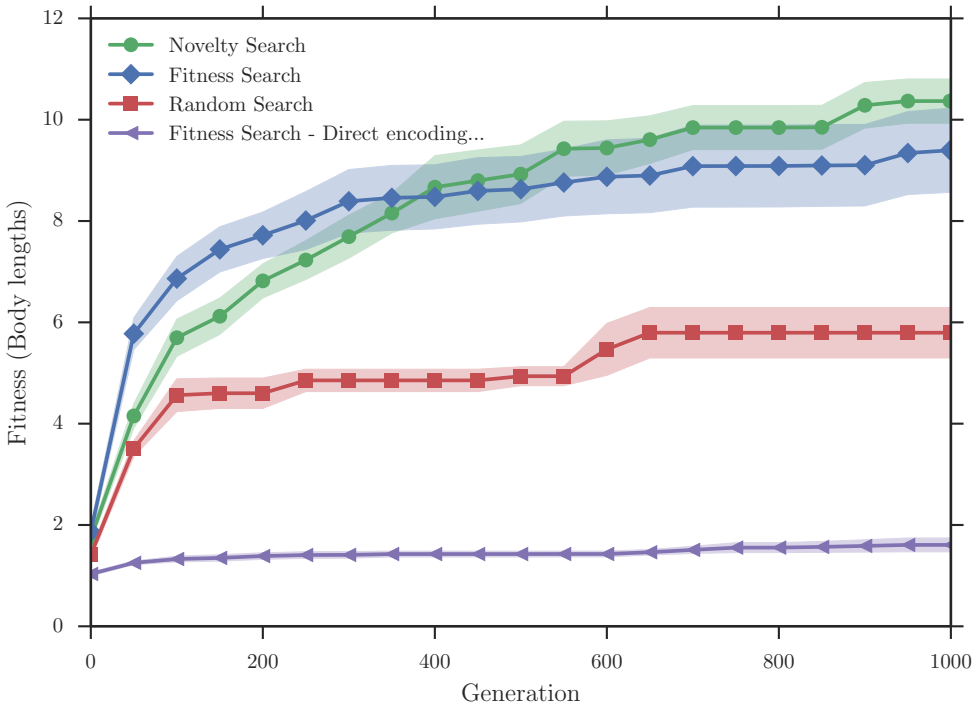


FIGURE 5.7: Comparison of simple genetic algorithm (direct encoding) against novelty-fitness-random search with generative encoding. Best fitness so far averaged over 10 runs. (Settings B.3)

have to deal with an optimization problem, where local optima and global ones can be found as the number of the possible solutions in this setting, using 4 materials, is $\sim 2,3 \times 10^{87}$. Using the two-dimensional trajectories of the soft robots, novelty search visits optimal solutions that none of the other methods does. Local optima can prevent fitness based search to achieve the performance of novelty search. Encoding limitations in direct encoding cannot lead to optimal solutions for this settings. In the case of random search, the individuals of each generation are selected randomly to reproduce. Having neither the information about their fitness, nor the driving force of novelty search that seeks for novel behaviors, it fails to evolve any decent locomotion. The only reason random search in CPPN-NEAT achieves to evolve displacement of ~ 5 body-lengths, is the powerful encoding used (CPPNs). The simple genetic algorithm approach which uses a direct encoded chromosome to represent the structure of the soft robots performs better than using random selection with an indirect encoding. Structural symmetry and regularity does not show all of its merits in such low resolution settings.

Moving to higher resolution lattices, it is expected that generative encoding will prove its advantages over the direct encoding scheme (Stanley, 2007; Cheney et al., 2013). More complicated morphologies can be produced (morphology space for 10^3 lattice resolution: 9.3×10^{698}). Furthermore, the space of behaviors, for instance two-dimensional

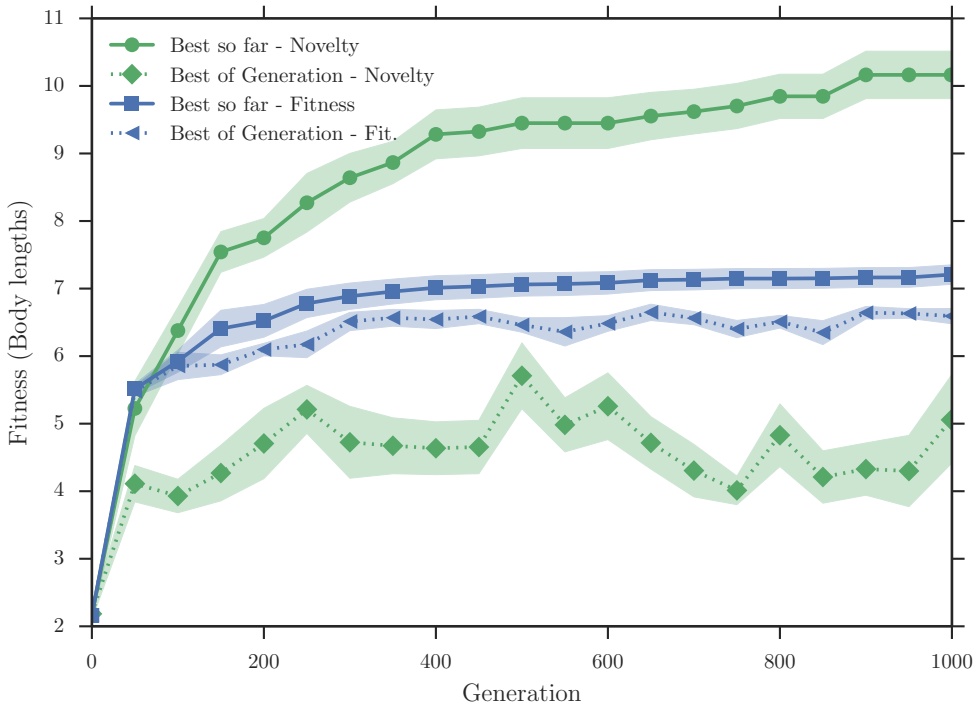


FIGURE 5.8: Fitness of the champion per generation alongside best fitness so far for fitness-novelty search, averaged over 10 runs. (Settings B.2)

trajectories, becomes larger since bigger and more detailed soft robots can achieve higher displacement and more complex gaits. As it has been shown before, these higher resolution morphologies can achieve life-like locomotion. The same experiment as before held under a lattice resolution of 10^3. Figure 5.7 illustrates the performance (i.e best displacement so far) of the four different methods in these higher resolution settings. Results reassure that novelty search achieves higher fitness on average against fitness based search. Nevertheless, there is no tremendous difference as in the previous experiment. Both methods achieve to evolve the soft robot structure with the highest fitness found in all experiments ($\sim$ 14 Body lengths). Novelty search behaves more constant in evolving individuals with high fitness in all runs, on the other hand most of individual runs of fitness search are being trapped in low fitness local optima, trying to optimize specific individuals without trying to explore deeply the fitness landscape like novelty search successfully does. Random selection within CPPN-NEAT evolution produced low fitness morphologies for soft robots. The high difference between random selection evolution and novelty search proves that seeking novel behaviors in novelty search cannot be considered as a random search. The superiority of generative encoding (CPPN) over direct encoding can be evidently observed. Regular in shape morphologies can take advantage of their geometrical properties to locomote efficiently. Moreover, the performance of direct encoding when a higher resolution lattice is used for the soft robots is

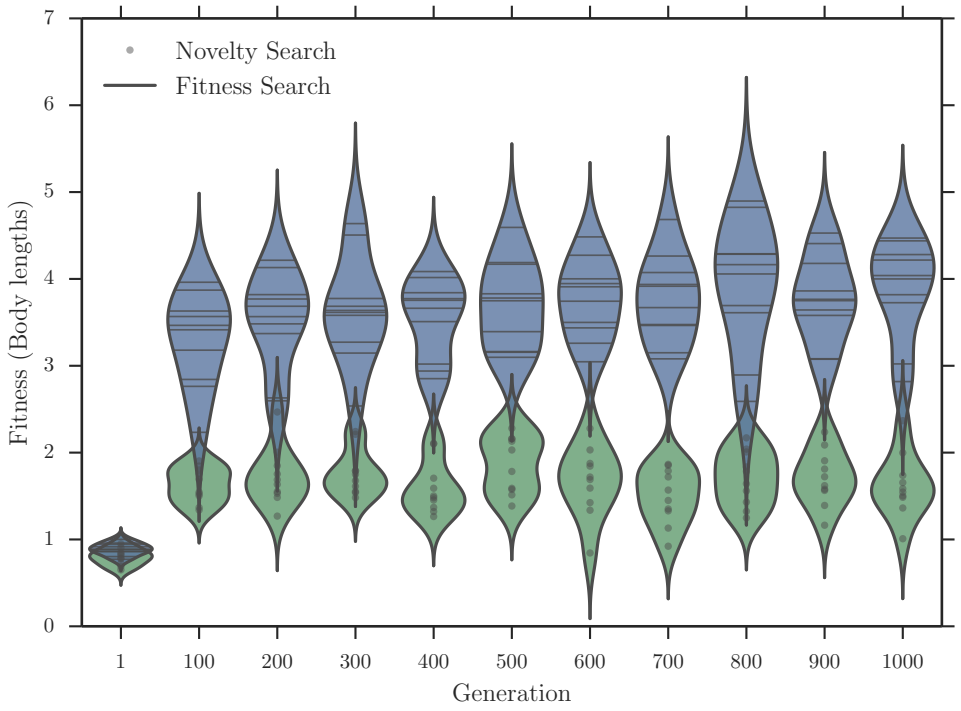


FIGURE 5.9: Distributions of the average fitness of the population every 100 generations, results from 10 runs of fitness (Blue) - novelty (Green) search with generative encoding. (Settings B.2)

radically decreased. Structure and morphological regularity is a necessity for soft robots in order to perform decently in this resolution, a property that direct encoding cannot capture failing in these settings.

Novelty search managed to improve the average displacement of the soft robot morphologies. The metric that both search methods are compared against each other, is the best fitness so far. Only the champions of each generation affect this metric while an important question is how the population of each generation is affected with regards to the search method that is being followed. The average fitness of the population, as well as the champion's fitness are two cues that can point out interesting properties of each method. Figure 5.8 presents the fitness of the champion soft robot (best within the generation) alongside the best fitness so far for both novelty and fitness based search averaged over 10 evolution runs. In fitness based search, champions of each generation are getting better through the evolution resulting to an approximately monotonically increasing function. However, in novelty search a random pattern for the champions of each generation can be observed. An early improvement is mainly caused by the generative encoding while the performance of the generations' champions is not affected by the search towards novel behaviors. What is interesting, is that on average the champions during novelty search evolution are worse than those fitness search evolves, whereas

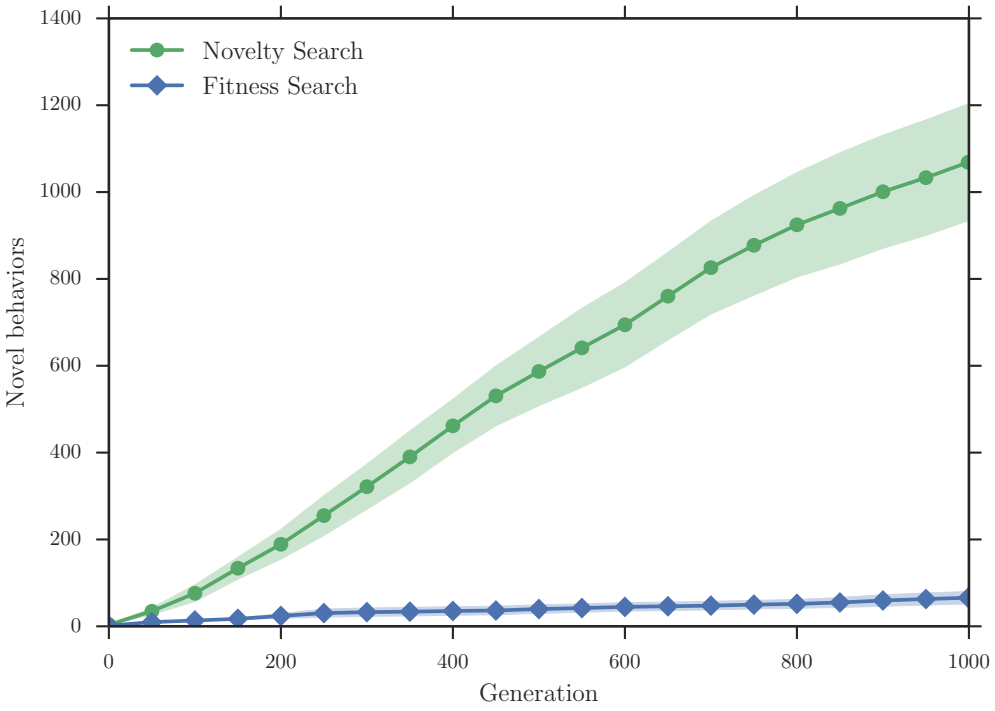


FIGURE 5.10: Number of novel behaviors found up to generation, averaged over 10 runs. The novelty measure is computed as the average distance from the 10-nearest behaviors for fitness-novelty search with generative encoding. (Settings B.2)

the best fitness so far is better for novelty search in this setting (lattice resolution 7^3). Hence, individuals that resulting in the increased performance of novelty search clearly lie on the tails of the fitness distributions on each generation. In the same fashion the average fitness of each generation also seems to be affected by the different optimization method. Figure 5.9 illustrates the distribution of the average fitness per generation of 10 independent runs for novelty and fitness based search. The average fitness of each generation is shown for every 100 generations. The violin-like distributions show that the average fitness per generation remains stable through the whole evolution (1000 generations) for both methods. Additionally, the average fitness is significantly lower for novelty search, meaning that when the evolution is being driven towards novel behaviors there is no guarantee that novel findings in the behavior space are also fit solutions. What has been shown in the last two figures (see Figs. 5.8,5.9), evidently shows that although novelty search achieves finding more "fit" solutions than fitness based search in the specific problem domain, the average fitness of both generation champions and population remain lower than in fitness based search.

Until this point, the performance of both fitness and novelty search methods have been compared in the same objective metric, the displacement of the produced soft robot morphologies. The former method tries to optimize genomes in respect to the specific

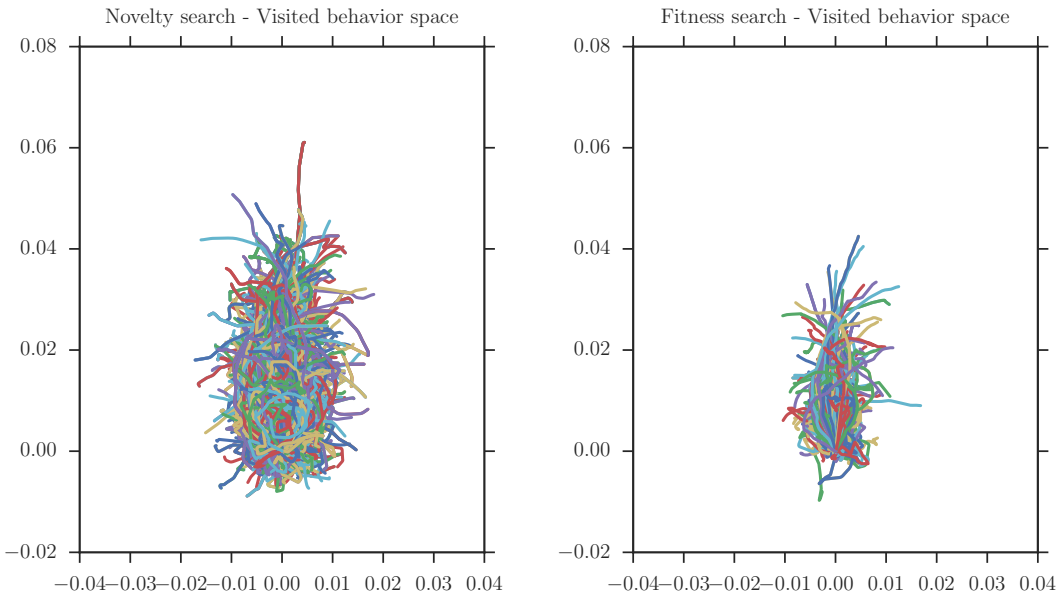
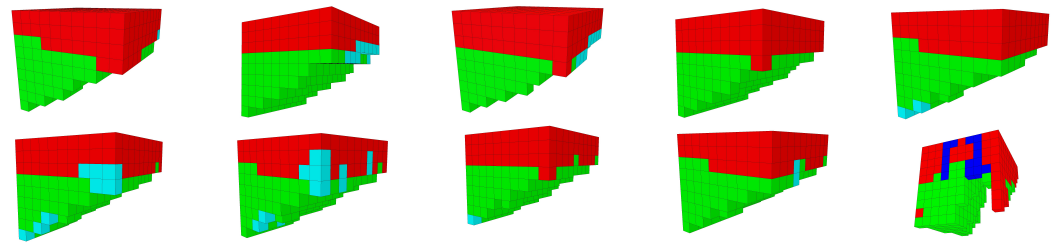


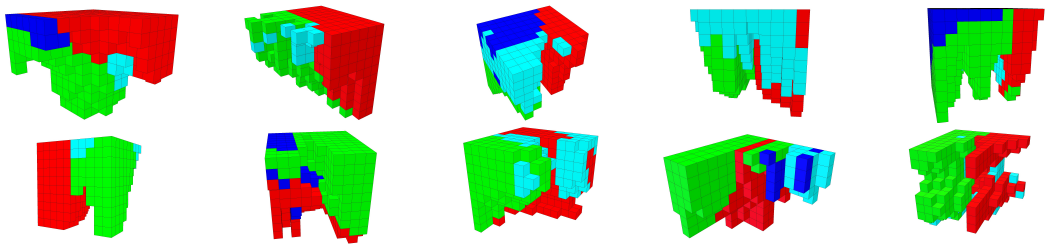
FIGURE 5.11: Novelty search visits a vast amount of behaviors achieving in this way to find fit individuals, and avoid local optima of the objective function. (Settings B.2)

objective function, while the latter is focusing the search into creating diversity of the population in the behavior level. It is of interest to show the performance of both methods in a different evaluation metric. The evaluation metric is used in this experiment is the number of novel behaviors generated within an evolution run. Inverting the evaluation metric, so it is in favor of novelty search, it is expected that novelty search will outperform fitness based search by a huge margin. Figure 5.10 presents the number of novel behaviors that the two evolutionary methods generated averaged over 10 runs. Both methods use the same settings (behavior/constant values) to determine the novelty of a behavior. The resulted plot shows that comparing these two methods in this metric is pointless as novelty search drives the evolution towards spaces in the behavior space that have not been visited before. As a result, novelty search produces approximately one novel behavior per generation. Novelty search achieves better performance than fitness based search in both objectives set so far, evolving fit, and at the same time diverse solutions. To visualize the difference in the behavior space of the two methods, Figure 5.11 illustrates all the stored novel behaviors (two-dimensional trajectories) found in one evolution run of novelty and fitness based search. The initial position of the soft robots is the start of the axes, and the centroid of the trajectories is normalized to be perpendicular to the horizontal axis. The differently colored trajectories verify that novelty search searches the space of behaviors more than fitness based search does. In addition, observing the performance of these trajectories is easy to interpret that longer trajectories (higher fitness) have been produced by the first method.

In this section, the performance and several aspects of novelty search has been discussed.



(A) Fitness based search



(B) Novelty search

FIGURE 5.12: Fitness based search trying to optimize a specific structure, whereas the search for novelty results in a variety of shapes. (Settings B.3)

Novelty-search achieved higher average displacement of the evolved soft robots against traditional fitness based search, where two-dimensional trajectories have been used to define the behavior in all settings (lattice resolutions of 5^3, 7^3, and 10^3).

5.2.1 Diversity of Individuals in Novelty Search

The gain in performance that novelty search achieved over fitness based search has been discussed in detail. At the same time, evolved morphologies illustrated earlier in this chapter (see Sec. 5.1) show that both methods can create morphologies that are able of efficient locomotion. The diversity of the individuals in the behavior space verified how novelty search can achieve the gain in the objective measure by seeking for novel behaviors. Another interesting question is what happens in the morphology of the soft robots during the evolution under these two different methods. Figure 5.12 shows the champions every hundred generations of an experimental run for novelty and fitness based search. While the fitness based search is focusing on the optimization of a specific morphology, novelty search is searching the behavior space unveiling novel behaviors and morphologies. The same motif appears in every independent run of fitness and novelty search. Novelty search evolves a larger variety of morphologies, whereas fitness based evolution is focusing to certain shapes, different in every run. Both search techniques have their advantages and disadvantages. First, fitness based search optimizes (optimized distribution of materials within the structure) certain shapes during the evolution, while novelty search does not optimize them. Althoug novelty search allows the evolution of highly novel behaviors-morphologies, these novel morphologies even in

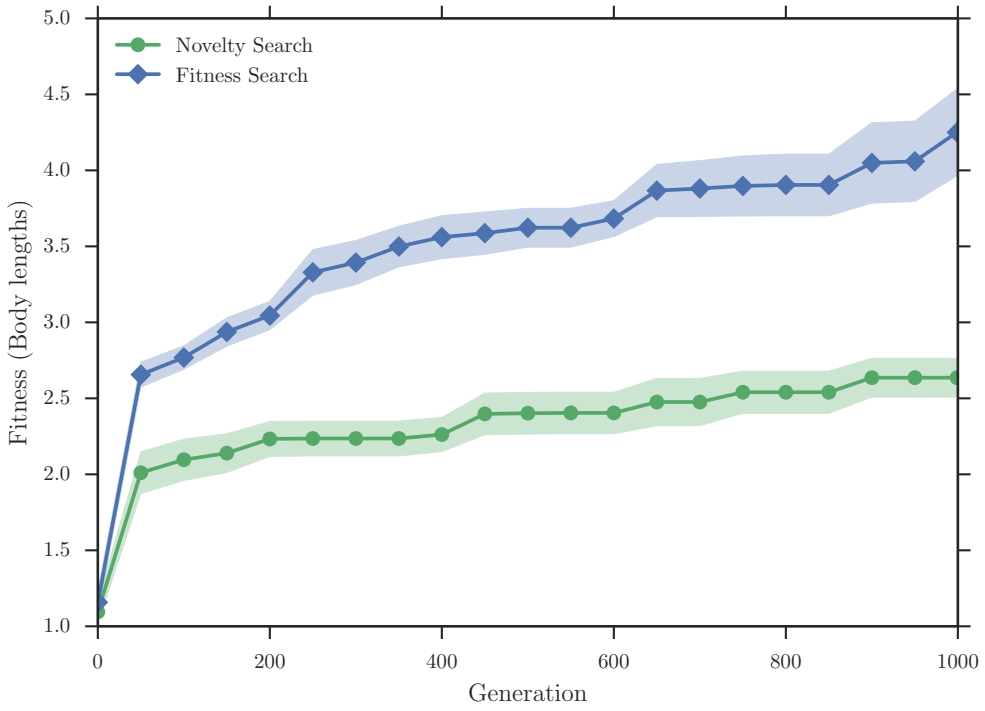


FIGURE 5.13: Best fitness so far penalizing actuated materials for fitness-novelty search with generative encoding, averaged over 10 runs. (Settings B.1)

cases that they perform well under a task will never have the time to be improved. Later in this Chapter ways of combining both search methods' advantages will be investigated.

5.2.2 How Behavior Selection Affects Novelty Search

In novelty search a good behavior metric must contain information about the objective function the search is trying to optimize the individuals for. In the case of evolving the gait of soft robots, trajectories can be highly informative regarding the displacement of the robot's body, as well as the locomotion strategy that it is observed. Two soft robots with different gaits, which traveled the same distance within an equal time horizon, have the same fitness if displacement is only measured. Most objective functions used in the evolution of robot-gait cannot describe the locomotion strategy produced by the robot controller. The observed behavior of a robot can contain this information. Novelty search can take advantage of a descriptive behavior metric, forcing the evolution to seek novel solutions in the behavior space. As a result, it can result in the evolution of ten times more novel behaviors than fitness based search (depending on the threshold and the behavior metric) (see Fig. 5.10). Fit individuals will be found as the behavior space is heavily searched. The importance of the behavior metric in novelty search is crucial in order for fit solutions to be evolved. Two-dimensional trajectories in the

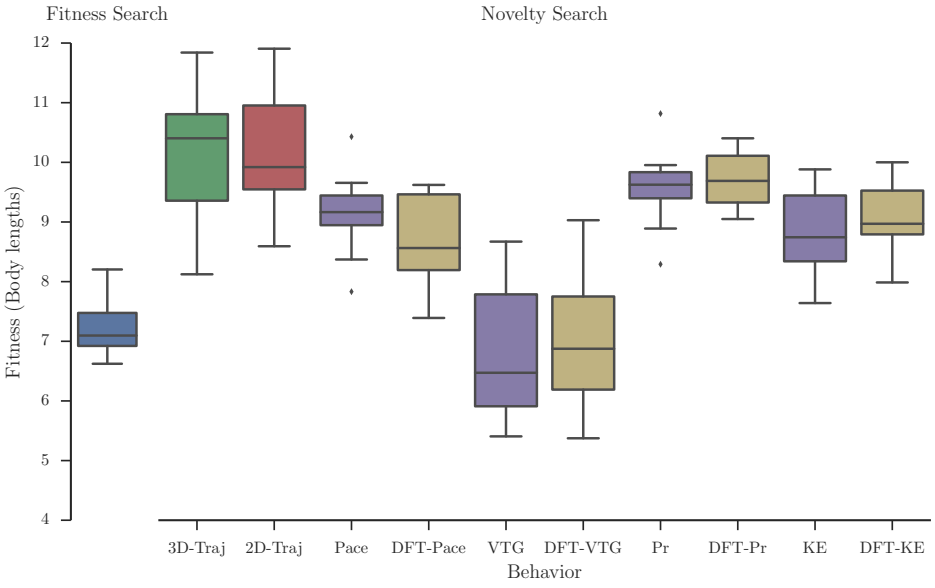


FIGURE 5.14: Distributions of the champion fitness resulted from 10 independent runs under different defined behaviors for novelty search. Fitness search is also evaluated under the same settings (left - blue box). (Settings B.2)

evolution of soft robots contain all the information needed to determine the fitness (displacement). This metric is incorporated inside the behavior (trajectories) assuming constant sampling rate of the trajectories. To prove that, the performance of novelty search is investigated when no information about the objective function is provided by the behavior, an objective function for which two dimensional trajectories do not contain information about is selected. The objective function is a penalized version[1] of the displacement of the soft robots in respect to the number of actuated (active) voxels in the soft robot. Figure 5.13 illustrates the best fitness so far for both novelty and fitness based search averaged over 10 evolution runs. Comparing the results with Figure 5.6 novelty search performs poorly in regards to the evaluation metric, whilst the same method outperforms traditional fitness based search when the whole information of the fitness function is incorporated into the behavior. Trying to find novel trajectories in the first case has been proven successful in respect to the final displacement of the evolved individuals, when the objective information was incorporated into the behavior. On the other hand, novelty search failed to optimize the distance that soft robots traveled and at the same time minimize the number of actuated voxels using the same two-dimensional trajectories as the behavior. As it has been shown, the performance of novelty search depends heavily on the selection of the behavior metric.

[1] Actuated materials penalized fitness function:

$$f = (1 - (n_{actuated}/n_{total})^{1.5}) \times disp$$

where $n_{actuated}$ is the number of actuated voxels, n_{total} total number of voxels, and $disp$ the displacement of the softbot's center of mass.

To verify the previous finding, where novelty search failed to compete with fitness based search when the behavior metric used did not contain information about the objective function, a set of behavior metrics are used. Figure 5.14 illustrates the performance achieved by novelty search for all behaviors defined in Table 4.1. In addition, the performance of fitness based search is presented in the left side of the figure (blue box). A set of 10 behavior metrics are used including the three-dimensional trajectories of the soft robots (3D-Traj), the two-dimensional projection on x, y-axes of the previous three-dimensional trajectories (2D-Traj), the pace of the soft robots sampled every 0.001 sec. (Pace), the discrete Fourier transformation of the same signal which is sampled every 0.00001 sec. (DFT-Pace), the number of voxels touching the ground on each sampling time-step (VTG, DFT-VTG), the maximum pressure per time-step (Pr, DFT-Pr), and the kinetic energy of the whole structure (KE, DFT-KE). What is shown in Figure 5.14 is the fitness in body lengths of the champion soft robot of the whole evolution from 10-independent runs. Both trajectory-type behaviors achieve the best performance with regards to the fitness measure. The fitness distribution of overall champions in novelty search with the two-dimensional trajectories shows a small difference in favor of two-dimensional over three-dimensional trajectories. The third highest performance is achieved by the maximum pressure behavior, which is close to the previous two trajectory-type behaviors. Pace and kinetic energy of the soft robots are the next best behavior-types in the performance ladder. The worst behavior metric regarding the fitness that is achieved is the number of voxels touching the ground behavior.

The performance of novelty search when trajectories of the soft robots are used as a behavior metric is superior over all other behavior metrics. Trajectories are a very good selection for this kind of evolution since they can indirectly encode not only the objective function which is the displacement, but also the locomotion strategy. The reason why they achieve such a high performance is that novel behaviors in the space of trajectories result in long trajectories which can be described novel. The rest of the behavior metrics apart from VTG and VTG-DFT are close as far as the final performance of the evolution is concerned. One reason they fail to meet the trajectories' performance is the fact that although they keep track of cues that can describe the performance of the robot (speed/displacement), they cannot encode the direction of them. Soft robots having a circle trajectory can produce fast locomotion, in this case though, the measured displacement from their initial position remains low. Counting the number of voxels of a soft robot that touched the ground in every sampling timestep of the simulation, does not imply how fast the robot is moving. A fast moving robot that is hopping can have a similar behavior with a hopping robot that stays in the same position after each jump.

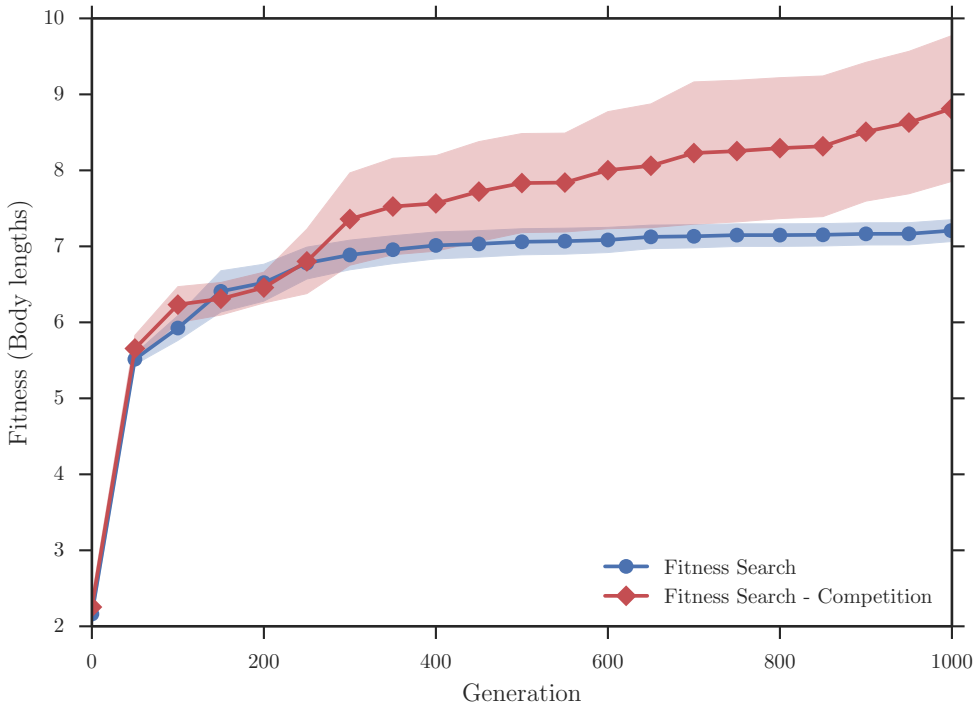


FIGURE 5.15: Best fitness so far with no competition, and local competition in the complete population of each species for fitness search, averaged over 10 runs. (Settings B.2)

On the contrary, using the trajectories of these two soft robots, the behaviors observed would have been highly variant.

In the same figure (see Fig. 5.14) and on the left side of it (blue box), fitness based search is also evaluated under the same experimental settings. The performance of this objective optimization method is comparable only to the novelty search search when the voxels touching the ground are the selected behavior metric. Apart from the effects of the behavior selection another aspect of novelty search, the sparsity equation is investigated in detail in Appendix D.

5.3 How Selection Affects the Performance of Both Search Methods

Discussed extensively in a previous Chapter (see Sec. 2.1.1), selection is a process that picks individuals in order to breed, be mutated, or be copied into the next generation. It is the part of any evolutionary algorithm that is responsible for producing the next generation based on the individuals which exist into the current one.

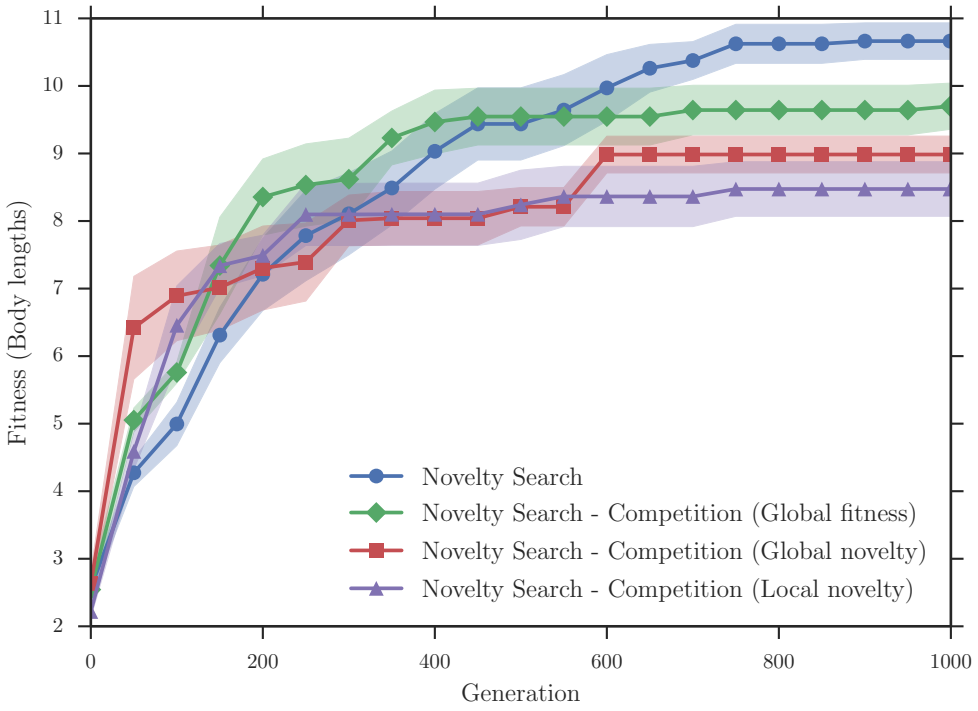


FIGURE 5.16: Best fitness so far, local competition inside each species for novelty search with generative encoding, averaged over 10 runs. (Settings B.1)

Fitness search

To discuss the effects selection method has in the specific evolutionary setting, competition is used in both fitness based and novelty search. Figure 5.15 presents the results for two different selection methods, random selection from the top 20% (Blue) of the population, and competition among individuals from the whole part of the current population (Red). The size of the competition used in this experiment was 4, meaning that for every genome to be produced its parents will be the best genomes in respect to their fitness value from 4-random picks within the current generation. Because the NEAT method uses speciation (see Sec. 2.4.1), the competition is held among species. As it is expected, competition and the fact that the whole population has the opportunity to breed contribute to the diversity of the population. This can be easily seen in Figure 5.15 where random selection within the top 20% of the population does not allow solutions to reproduce, meaning that it does not explore weaker individuals which may have the potential to become better after a decent number of mutations or cross-overs with other individuals. The deviation of the first method gives a perfect clue about how narrow is the fitness landscape at the converged area of search when only the best solutions survive on each generation.

Novelty search

Since the algorithmic framework is the same for both search methods, competition can also be used in novelty search. Figure 5.16 presents the results when competition is held among individuals of the whole generation's population within the same species. Competition is held among individuals in respect to their global (when they are compared to all novel solutions) and local (when they are only compared to their species' population of novel behaviors) novelty, red, and purple lines respectively. In both cases the overall performance of the evolution averaged over 10 runs is worse than pure novelty search method where individuals are selected randomly from the top-20% of the population. Both selection methods, set aside an early gain in the evolution, are performing poorly set when compared to the default selection method. Selecting individuals with high novelty within the species is crucial for the reduced performance since these individuals can have low novelty value when compared with the global population leading to steps backwards in the evolution towards novel individuals. On the other hand, when individuals are competing using their global novelty measure leads to a slightly better performance, still far from the default setting. On the contrary to the fitness based method where competition achieved an improvement, competition within novelty search was not successful.

Competition allows individuals to breed although they are not in the top part of the population in respect to an evaluation measure. For fitness based search this means that soft robots with low displacement will be allowed to breed and some of them eventually will become better. For novelty search, it is less probable that behaviors with low novelty (behaviors located in a dense area of observed behaviors) will contribute in generating novel behaviors in future generations. Hence, allowing low novelty solutions to survive harms the performance of novelty search.

5.4 Incorporate Fitness Information into Novelty Search

The reason that novelty search is considered such a revolutionary search method is because it finds solutions for deceptive problems, where the fitness landscape is not a straightforward function. What makes it so unique is the fact that instead of looking for optimizing the solutions in respect to an objective function is looking for the novelty in the behavior space. On each generation of novelty search novel behaviors that are also fit in regards to the objective of the problem are discovered. Mutations of these solutions will yield in behaving similarly to their ancestors, resulting in similar behaviors. Thus, the novelty value of these individuals will be declined as similar behaviors will

contribute in a denser area in the behavior space. Eventually these solutions will stop being selected, and evolution will not have the chance of carrying their valuable genes along. Mutations and other genetic operations can optimize these fit individuals more. These individuals (with high fitness value) can be seen as *stepping stones* (Lehman and Stanley, 2011a) towards more optimized versions of them. Being blind to the objective function, novelty search will eventually stop producing new individuals out of them, which will lead to promising individuals being unable to survive through the evolution process.

Fitness-competition in novelty search

Competition is a simple way of combining these two search methods together. In this experiment the number of new individuals each species will breed is determined by the average novelty value of each species. However, competition is selecting these individuals not in respect to their novelty but regarding their fitness value. After each generation is produced, competition is held over all the population within each species to select individuals for reproduction. Figure 5.16 illustrates the results of using the fitness of an individual as a measure for selection among two generations. The resulted best fitness so far (Green line) reveals that competition for fitness in a novelty search setting disturbs the balance of the evolution towards novelty. Competition in respect to fitness is not allowing novelty search to expand the search in the behavior space in a greater extend since it is not the case that selected fit individuals will lead in novel behaviors.

Fitness-elitism in novelty search

It has been shown how selecting individuals in respect to their fitness by competition leads to a declined performance for novelty search. Hence, an approach is proposed for incorporating fitness information into novelty search without perturbing with its pipeline. Elitism is the process of passing mutations or copies of the best individuals to the next generation. In this way best individuals are preserved and can be optimized later. The best individuals of each species generation are protected so they can contribute with their beneficial genes later in the evolution. Novelty search can include elitism in its selection process, and it does that by copying the most novel organisms of the current population of each species to the next. Since there is no point of changing this function, elitism can be used also to copy fit individuals within novelty search method. The way these two elitism functions can be combined together depends on the population size and the problem, while probabilistic methods can also be used. In the specific setting, both elitism function copy new individuals to the new generation with

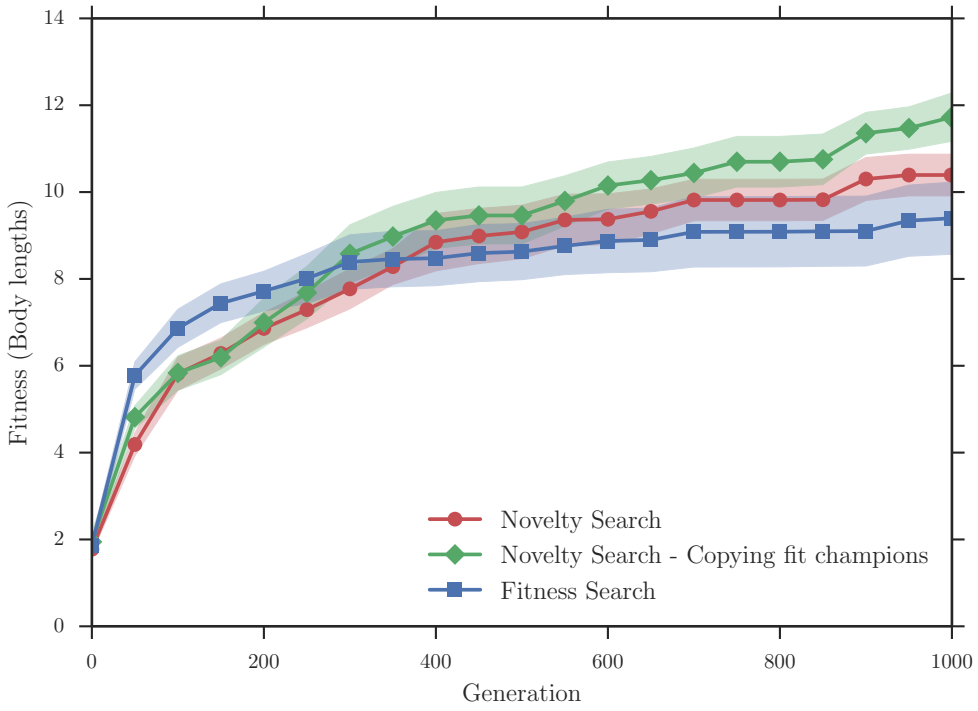


FIGURE 5.17: Best fitness so far, novelty search with and without copying *fit* champions, and fitness search, averaged over 10 runs. (Settings B.3)

probability one. Moreover, evolution towards novelty does not get disturbed, at the same time fit individuals have the chance to be optimized further as long as they are the fittest within the species population. Figure 5.17 illustrates the gain in performance when fitness elitism is used in novelty search method compared with the pure novelty and fitness based search methods.

In this section two ways of incorporating fitness information into novelty search method investigated and discussed. Competition and elitism can be used within the pipeline of novelty search in an evolutionary algorithm to use fitness information of individuals. The latter achieved to an improvement over pure novelty search method.

5.5 Evolving Soft Robots for Outer Space

In this section, it is of interest to show how different environmental conditions can affect both the performance and the type of locomotion produced by the evolved soft robots. Both search methods discussed in this thesis, novelty and fitness based search, are used for the co-evolution of the morphology and the locomotion strategy of soft robots under variant gravity levels. Since the software used for the simulation of the soft robots cannot fully reproduce the conditions of other planets/moons in our solar

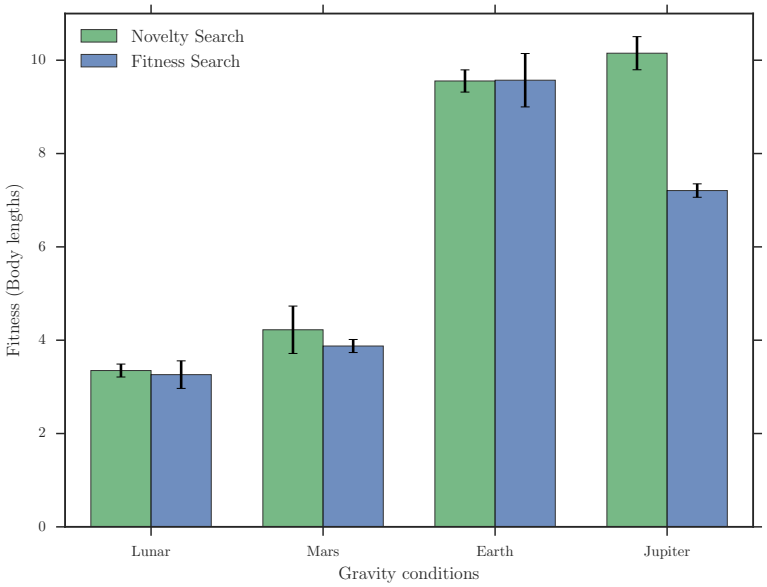
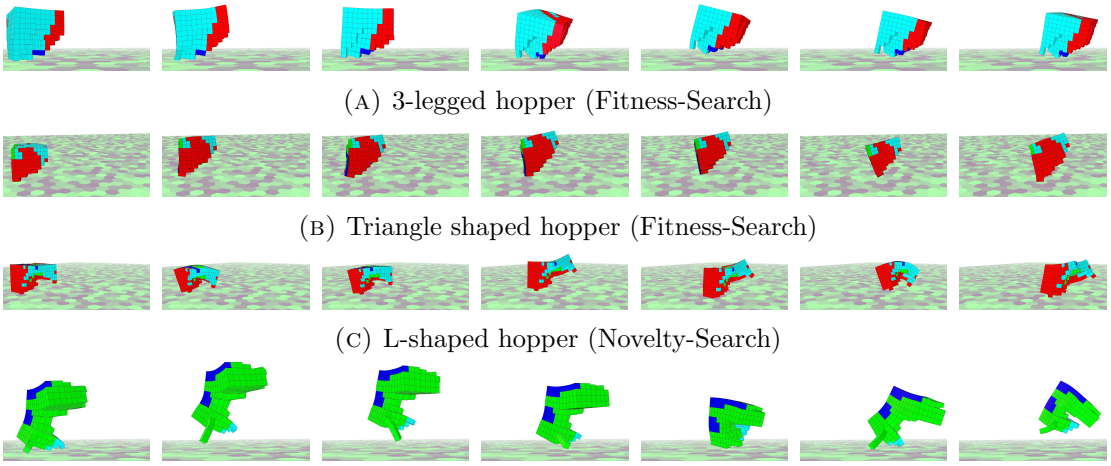


FIGURE 5.18: Novelty search performs better or equally good than fitness based search in all gravity conditions tested. (Settings B.8)

system, the gravity acceleration of the environment is the only altered variable of the simulation environment. For the novelty search method two-dimensional trajectories of the soft bodies are chosen as the behavior metric to evaluate the novelty of each individual. Due to the computationally expensive simulation for structures of resolution 10^3, the performance of the locomotion strategies evolved is measured in a smaller lattice resolution 7^3 (10-runs for each gravity level for both methods). Settings used in all previous experiments were also used for Jupiter's and Earth's gravity accelerations, the simulation time used for both was 0.4 seconds. For Lunar's and Mars' evolution runs a higher temperature period was used (0.050 instead of 0.025 seconds), in order effective locomotion to take place. High frequencies tend not to allow soft body structures produce any decent locomotion in lower gravity conditions. Furthermore, for the lower gravity of Lunar and Mars, the simulation time was increased up to 1 second for each evaluation. However, the final displacement of the robots was normalized to meet the expected displacement for 0.4 seconds (the displacement of soft robots is roughly linear to time).

Figure 5.18 illustrates the performance of novelty and fitness based search in four different gravity conditions for Lunar ($-1.6\ m/s^2$), Mars ($-3.7\ m/s^2$), Earth ($-9.7\ m/s^2$), and Jupiter ($-24.8\ m/s^2$). The best fitness achieved by an individual averaged for all runs are shown together with the deviation errors. Novelty search achieves in producing better or equally good locomotion for the soft robots evolved in all gravity conditions. The results on different settings verify that novelty search can indeed achieve higher performance in the specific problem task.



(A) 3-legged hopper (Fitness-Search)

(B) Triangle shaped hopper (Fitness-Search)

(C) L-shaped hopper (Novelty-Search)

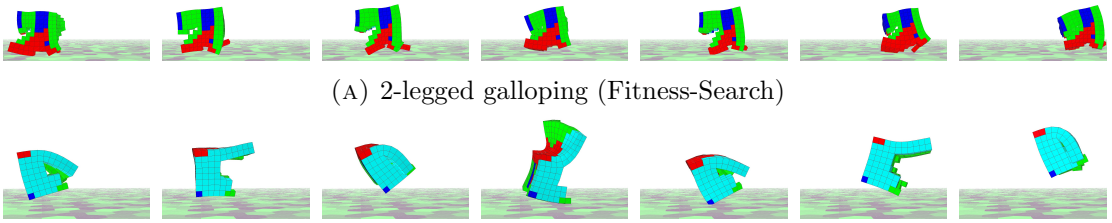
(D) C-shaped hopper (Novelty-Search)

FIGURE 5.19: **Lunar**: Locomotion strategies evolved in low-gravity conditions of Lunar consist mostly of hopper soft robots. (Settings B.4)

The rest of this section discusses in detail findings during the experiments in regards to the evolved locomotion strategies and morphologies capable to produce efficient locomotion. Although the measured performance of 10 evolution runs can give us a clear estimate of what each method can achieve in respect to the expected displacement of soft robots, the resolution of 7^3 is not high enough to unveil complex morphologies. Three additional experiments were performed at a higher resolution (10^3), revealing morphologies and locomotion strategies observed by the evolved soft robots. However, the number of runs was not high enough to give a reliable estimate of the performance with regards to the displacement of the soft robots.

5.5.1 Soft Robots on Lunar

Locomotion strategies evolved under low gravity conditions for the gravity of the Lunar more specifically, showed that only hopping gaits can produce effective locomotion in these conditions. Low gravity makes it difficult for the soft body structures to grip on the ground surface and evolve something different than hopping. Figure 5.19 shows four different types of hopper soft robots under Lunar's gravity. However, the morphology of each hopper differs. A 3-legged hopper (see Fig. 5.19a) uses its leg in the middle to hop, stabilizing itself with the help of its front and back legs. One legged hoppers can be evolved with different morphologies, a triangle shaped body (see Fig. 5.19b), L-shaped (see Fig. 5.19c), and a C-shaped soft robot (see Fig. 5.19b). Hopper soft robots are difficult to develop a stable locomotion strategy since most of the times the hopping technique they use fails after few simulation seconds. Moreover, stable locomotion is difficult to be evolved, since the evolution objective is the locomotion speed.



(A) 2-legged galloping (Fitness-Search)

(B) 2-legged C-shaped hopper (Novelty-Search)

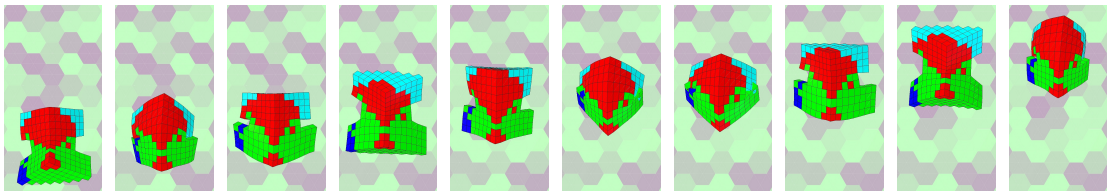
FIGURE 5.20: **Mars**: Gravity acceleration on Mars allows both galloping and hopping locomotion strategies. (Settings B.5)

5.5.2 Soft Robots on Mars

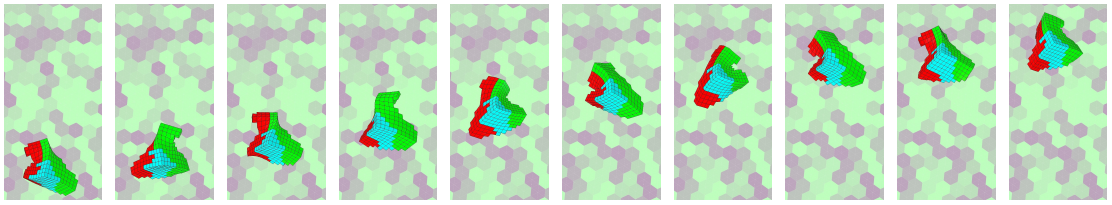
The locomotion effectiveness on Mars is higher when is compared to this on Lunar's gravity acceleration making it possible for the virtual soft robots to evolve other kind of gaits using legged bodies. Figure 5.20 presents two evolved virtual creatures where the one is galloping having a two-legged body (see Fig. 5.20a), and the other is hopping having a C-shaped soft body (see Fig. 5.20b). Note that the C-shaped hopper soft robot mostly uses passive materials apart from its upper body where all the active material are located. With using its upper part generates enough motion able to move itself. What is observed in the morphologies of soft robots evolved in lower gravity levels was that the use of less number of active voxels can produce decent locomotion.

5.5.3 Soft Robots on Earth

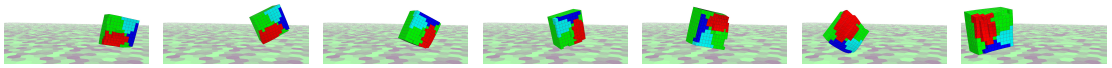
On higher gravity levels life-like locomotion emerges. Figure 5.21 shows three different locomotion strategies generated by fitness based and novelty search on gravity conditions of Earth. Galloping-type locomotion is again observed by evolved two-legged shaped body creatures (see Fig. 5.21a). Interesting animal-like gait has also been evolved (see Fig. 5.21b) verifying the connection there is between gravity and the locomotion strategies of living organisms evolving on Earth for thousands of years. Tumbleweed-like locomotion (see Fig. 5.21c) has been emerged under novelty search method producing rolling soft robots that can locomote efficiently. Fact that adds significance to the novelty search method since fitness based search did not produce this kind of locomotion strategy. Tumbleweed is a concept of low-cost exploration that has inspired robot designers for Mars' missions in the past (Antol et al., 2003) and has been already deployed in Antarctica for testing purposes by NASA.



(A) *Top view*, 2-legged galloping (Fitness-Search)



(B) *Top view*, 4-legged animal like locomotion (Fitness-Search)



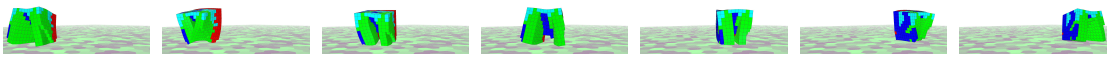
(C) Tumbleweed-like locomotion (Novelty-Search)

FIGURE 5.21: **Earth**: Morphologies evolved in gravity conditions on Earth show that life-like locomotion strategies can be generated by soft body creatures in a simulated environment. (Settings B.6)

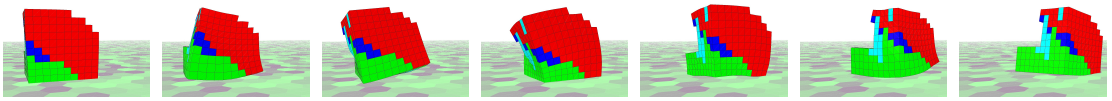
5.5.4 Soft Robots on Jupiter

Moving on to higher gravity levels, Jupiter, heavier structures can use galloping as a strategy for their locomotion. Figure 5.22 presents some of the locomotion types when novelty and fitness based search were used for the evolution of them. Galloping (see Fig. 5.22a) is again considered to be an effective way of moving in such a high gravity, whereas thicker legs are evolved to withstand the high gravitational force. Push-pull worm-like locomotion (see Fig. 5.22b) can also produce decent velocities for soft robots. Finally, hoppers have also been evolved to this setting, while they are using more actuated materials.

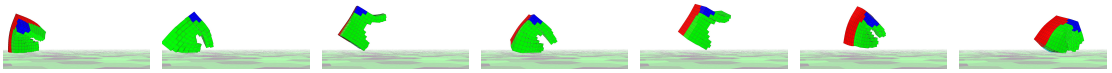
Different locomotion strategies can be evolved on different gravity levels producing effective locomotion. Low gravity does not allow other kinds of locomotion apart from hoppers to be evolved, while higher gravity acceleration allows more complicated behaviors to be evolved. In all settings, both search methods produced effective locomotion for the soft body structures, however, the performance in regard to the objective measure defined, displacement of the body in body lengths, was equal or higher for novelty search in all gravity settings.



(A) 2-legged walking (Fitness-Search)



(B) Push-pull locomotion (Fitness-Search)



(C) C-shaped hopper (Novelty-Search)

FIGURE 5.22: **Jupiter**: Heavier structures on Jupiter's gravity level can locomote efficiently using several strategies. (Settings B.7)

Chapter 6

Conclusion

In this chapter the contributions of this thesis are discussed. The simultaneous evolution of morphology and locomotion strategy of soft robots by novelty search has been investigated in depth.

Neuroevolution of augmented topologies evolutionary algorithm was used to evolve compositional pattern-producing networks, as the chromosome representation. CPPNs can be queried for the lattice dimension of VoxCad simulation software and output not only the morphology but also the distribution of materials in the structure of a soft robot, determining the locomotion strategy that will be generated. The merits of using a generative encoding scheme like CPPNs were shown, where regular patterns in the output of these artificial networks exploited the properties of the problem task. This generative encoding showed its advantages in the case that random selection used in the evolution, whereas random generated soft robots and direct-encoded soft robots could not produce any decent locomotion strategy.

For the first time in evolutionary soft robotics a diversity based method such as novelty search was used, resulting in an unexpected and valuable scientific result. Novelty search method outperformed traditional fitness based search in evolving soft robots morphologies that can move fast in a virtual environment. Both techniques compared under the same objective function which was the normalized by body-length displacement of soft robots within a fixed time-span. All the experiments presented throughout this thesis proved that the search towards the above evaluation metric mentioned was not as successful as deploying novelty search at the exact same settings, driving the search towards a diversity in the behavior level. The resulted performance of novelty search method in this setting showed that seeking for diversity in the behavior space can result in an improved performance in the fitness metric, which metric traditional methods optimize their population for. The effect that behavior selection has in the evolution of

the morphology and the effectiveness of locomotion strategy of soft body structures was investigated in detail. Several behavior metrics designed and deployed in novelty search showing the performance difference when each one was used to define the behavior space. Moreover, it was shown that a good behavior metric must contain information about the objective function which is subject to the optimization of the problem task. Previous work in evolving virtual creatures by novelty search (Lehman and Stanley, 2011b) used the resulted morphology of the robots created to determine the novelty of an individual. The resulted performance for pure novelty search method was worse than the fitness based. Here, we verified that novelty search cannot perform equally good or better to fitness based search when the objective function is not well defined. On the contrary, well defined behavior metrics can lead novelty search to outperform traditional fitness based search in these settings. The obtained diversity in the phenotype level, and the role of sparsity were also investigated. Novelty search not only improved the performance and the diversity in the behavior space, but also contributed to the larger variety of virtual creatures evolved.

Additionally, different selection techniques used in both novelty and fitness based search, followed by a discussion on how this intermediate step in an evolutionary process can influence both search methods. The performance of fitness based search was improved when competition was chosen as an intermediate step between generations. Competing individual in regards to their fitness value allowed the evolution to exploit the fitness landscape better than in the pure fitness method. In addition, competition was used in novelty search method, where competition in respect to global and local novelty between each species disturbed the properties of the novelty search method, resulting in worse performance than the pure novelty search. Competition within novelty search was applied in regards to the fitness of the individual solutions. Once again, this method was not as good as the pure method. Another method of incorporating fitness information in novelty search was also proposed resulting in a significant performance gain over pure novelty search method. This was achieved through fitness elitism. Most fit individuals are passed through generations carrying their valuable genes and giving the chance to the evolution to benefit from the mutations or the crossovers with these fit solutions in future generations.

Finally, both techniques were used to evolve soft robots in four variant gravity levels, showing interesting results and the possibility of influencing future robotic designs for planetary exploration. Experiments under these gravity levels verified that novelty search is indeed performing better than fitness based search in the evolution towards high-velocity soft robots in this setting. In addition, some interesting characteristics of evolved locomotion strategies under variant gravity condition were also observed, adding knowledge and more possibilities when the gait of a soft robot under low or high

gravity is considered. With the progress three-dimensional printing is showing, future space missions can benefit from low cost soft robot explorers evolved to produce efficient locomotion. Passive locomotion can add more value to these soft body structures, where environmental variable conditions can actuate certain material types to produce locomotion.

Novelty search achieved higher performance in all settings used over fitness based search. However, assumptions about its generalized performance in evolving the morphology and the locomotion strategy of soft robots out of the simulation software used cannot be made.

6.1 Future Work

This section discusses possible future research that can be done in the topics approached and the contributions of this thesis. Designing soft body structures in a simulated environment is a heavy task for designers (Cheney et al., 2013), evolutionary algorithms (Stanley and Miikkulainen, 2002) and generative encoding (Stanley, 2007) combined, succeeded in the evolution of these designs, as well as the coordination (distribution of materials) of the structures evolved. What follows in this chapter is points worth further investigation in the science of evolutionary soft robotics, still in a simulated environment.

Evolution of materials

The scope of this thesis included the evolution of soft robots morphologies given a set of predefined materials with specific properties. The reason behind this, is the fact that it is was only of interest to investigate ways of designing soft robots having specific materials as building blocks. Another aspect in the evolution of the morphology of soft robot bodies is the evolution of the materials alongside the structure. The possibility of a dynamic palette of materials will enable more complex gaits for the soft robots evolved. Using the same generative encoding, material properties can be added as output nodes in the genotype representation (CPPN), resulting in a palette of materials which size is the same as the number of voxels presented in the evolved structure. Another possible way of evolving these properties could be that two genotypes can represent the same individual following two different encoding schemes. Direct encoding can be used for the material properties, whereas, mutations and crossovers will then only be held among genotypes of the same type.

Novelty search

Incorporating fitness information into novelty search proved to be profitable for this diversity rewarding search, adopting some of the advantages of fitness based search . Fit individuals can be selected at the selection process resulting to their ability to survive throughout the generations and be able to optimize their chromosome further. Although, a method that achieved in a significant gain over pure novelty search method was proposed, more ways of using genetic selection techniques can be used to achieve similar results.

As far as behaviors are concerned, a limited behavior space can benefit the search for novelty. Defining a valid behavior space and rewarding only behaviors within the valid space for their novelty. This technique used in (Lehman and Stanley, 2011a) is called *Minimal Criteria Novelty Search* and it is a way of making the behavior space more compact so only "good" behaviors are rewarded for their novelty. Doing so, novelty search incorporates indirectly further fitness information. In this thesis, the space of behaviors was only normalized for trajectories of the robot bodies, whereas the orientation of their displacement had no role to the novelty search. A limited trajectory space could only take into consideration straight trajectories treating all others as invalid behaviors and thus, not rewarding the individuals from which the trajectories were observed. It is expected that this type of novelty search will result in better solutions (Lehman and Stanley, 2011a) as the diversity of locomotion patterns will only appeal to the strategy and not the direction. It would be very interesting to see how minimal criteria novelty search could perform in this setting.

Objective function in the evolution of soft robot locomotion

The objective function defined by previous work (Cheney et al., 2013) and used throughout this thesis is the normalized displacement of the soft robot bodies within a limited time-span. Optimizing morphologies to achieve the highest displacement possible is resulting to emerged morphologies that are able to move fast but not stable enough to be considered good. An objective function that could contain information about the stability of the soft robots would have been crucial to the evolution towards more stable gaits. Furthermore, this stability measure would have been important in the evolution of locomotion strategy in lower gravity condition where hopping gaits were not stable.

Appendices

Appendix A

Simulation Settings

A.1 Environment

Settings used in the VoxCad simulation environment are given in table A.1.

TABLE A.1: Voxelyze simulation settings

Property	Value	Description
DtFrac	0.9	The timestep of the simulation, currently 0.9 × *dt*, where *dt* is the optimal timestep.
ColSystem	3	Hierarchical collision detection between all voxels. Updates potential collision list only when aggregated motion requires it[1]
StopConditionValue	0.4	Time in seconds simulation is stopped.
TempBase	25.0	Base temperature of the environment.
TempAmp	39.0	Temperature's amplitude of the environment.
TempPeriod	0.025	Period of the temperature cycle.
VoxelDim	0.001	Voxel dimensions, each voxel has length, height, and depth of 1mm.

A.2 Materials

In this section all materials' properties used during the simulations will be given. All materials used in the simulations have a set of shared properties which are shown in table A.2. Furthermore, unique characteristics of the materials are presented in table A.3.

[1] From VoxCad's documentation (Hiller and Lipson, 2012a).

Table A.2: Universal material properties

Property	**Value**	**Description**
Poisson's ratio	0.35	It is the ratio of expansion over two other axes following the compression in one.
Density	1×10^6 Kg/m^3	Density of material.
Temp phase	0	Phase of material to temperature period.
Static friction coef.	1	Static friction coefficient.
Dynamic friction coef.	0.5	Dynamic friction coefficient.

Table A.3: Unique per material properties

Name	**Color**	**Elastic Modulus** (MPa)	**CTE** ($1/deg\ C$)
Active positive (+)	Red	10	+0.01
Active negative (-)	Green	10	-0.01
Passive soft	Cyan	10	0.00
Passive hard	Blue	50	0.00

Appendix B

Experimental Settings

In this section the settings used for each experiment will be presented. For all the following experimental constants the simulation and material settings used are the ones described above, in case of other settings used, the new settings will be mentioned.

B.1 Lattice dimension 5^3

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Gravity acceleration $-27.6\ m/s^2$

Voxel dimensions $1mm \times 1mm \times 1mm$

Lattice dimensions $5mm \times 5mm \times 5mm$

B.2 Lattice dimension 7^3

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Gravity acceleration $-27.6\ m/s^2$

Voxel dimensions $1mm \times 1mm \times 1mm$

Lattice dimensions $7mm \times 7mm \times 7mm$

B.3 Lattice dimension 10^3

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Gravity acceleration $-27.6\ m/s^2$

Voxel dimensions $1mm \times 1mm \times 1mm$

Lattice dimensions $10mm \times 10mm \times 10mm$

B.4 Lattice dimension 10^3-Lunar

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Gravity acceleration $-1.622\ m/s^2$

Voxel dimensions $1mm \times 1mm \times 1mm$

Lattice dimensions $10mm \times 10mm \times 10mm$

B.5 Lattice dimension 10^3-Mars

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Gravity acceleration $-3.711\ m/s^2$

Voxel dimensions $1mm \times 1mm \times 1mm$

Lattice dimensions $10mm \times 10mm \times 10mm$

B.6 Lattice dimension 10^3-Earth

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Gravity acceleration $-9.780\ m/s^2$

Voxel dimensions $1mm \times 1mm \times 1mm$

Lattice dimensions $10mm \times 10mm \times 10mm$

B.7 Lattice dimension 10^3-Jupiter

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Gravity acceleration $-24.790\ m/s^2$

Voxel dimensions $1mm \times 1mm \times 1mm$

Lattice dimensions $10mm \times 10mm \times 10mm$

B.8 Gravity Experiments

Objective function Displacement in body lengths (displacement divided by size of soft robot) of soft robot's center of mass.

Table B.1: Unique per material properties

Planet	**Dim.**	**Grav.** (m/s^2)	**Sim. Time** (Secs.)	**Temp. Period** (secs.)
Lunar	7^3	-1.622	1.0	0.050
Mars	7^3	-3.711	1.0	0.050
Earth	7^3	-9.780	0.4	0.025
Jupiter	7^3	-24.790	0.4	0.025

Appendix C

Evolution Settings

Table C.1, presents the settings used in the evolutionary algorithm (CPPN-NEAT), the size of the population and the maximum number of generations are selected to match (Cheney et al., 2013) for comparison purposes. The size of the competition used in the some experiment is 4. For novelty search, the nearest neighbor sparsity equation was used for the 10-closest neighbor behaviors, at the same time the threshold in all novelty search experiments used was tuned so ~ 0.8 behaviors per generation are generated.

Table C.1: CPPN-NEAT settings

Property	**Value**
PopulationSize	30.0
MaxGenerations	1000.0
DisjointCoefficient	2.0
ExcessCoefficient	2.0
WeightDifferenceCoefficient	1.0
CompatibilityThreshold	6.0
CompatibilityModifier	0.3
SpeciesSizeTarget	8.0
DropoffAge	15.0
AgeSignificance	1.0
SurvivalThreshold	0.2
MutateAddNodeProbability	0.03
MutateAddLinkProbability	0.05
MutateLinkWeightsProbability	0.8
MutateOnlyProbability	0.25
MutateLinkProbability	0.1
SmallestSpeciesSizeWithElitism	5.0
MutationPower	2.5
AdultLinkAge	18.0
ForceCopyGenerationChampion	1.0
GenerationDumpModulo	1.0
ExtraActivationFunctions	1.0
SignedActivation	1.0
ExtraActivationUpdates	9.0

Appendix D

Additional Experiments

D.1 Sparsity in *Novelty*-Search

Sparsity (see Eq. 2.2) is a metric that evaluates how sparse is the space of a newly generated-observed behavior. In this equation k defines how many of the closest neighboring behaviors are going to be used for the computation of the average distance. Figure D.1 presents the resulted best fitness so far given different values for $k \in \{1, 2, 5, 10, 20\}$.

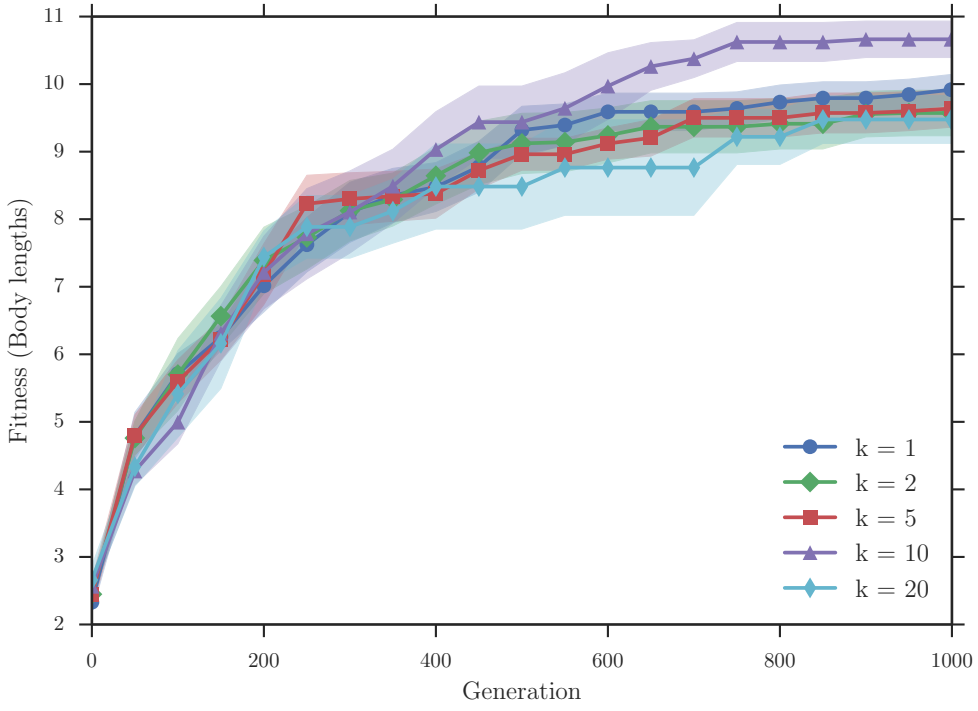


FIGURE D.1: Best fitness so far averaged over 10 runs, for different k to sparsity computation of the behavior. (Settings B.1)

In principle k can define how tolerate the algorithm can be with new behaviors. It is not certain that a specific value for k should give the highest performance in fitness and it depends almost completely by the application. The only implication in choosing value for k is that choosing large values should yield in a more detailed exploration in the behavior space. In the contrary, using small values the final set of behaviors will be denser in the behavior space. In the specific figure using $k = 10$ was the number of closest neighbors that led to the best performance.

Bibliography

Kenneth O Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic programming and evolvable machines*, 8(2):131–162, 2007.

Joel Lehman and Kenneth O Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation*, 19(2):189–223, 2011a.

Michael T Tolley, Robert F Shepherd, Bobak Mosadegh, Kevin C Galloway, Michael Wehner, Michael Karpelson, Robert J Wood, and George M Whitesides. A resilient, untethered soft robot. *Soft Robotics*.

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

Gregory S Hornby and Jordan B Pollack. Evolving l-systems to generate virtual creatures. *Computers & Graphics*, 25(6):1041–1048, 2001.

Gregory S Hornby, Hod Lipson, and Jordan B Pollack. Generative representations for the automated design of modular physical robots. *Robotics and Automation, IEEE Transactions on*, 19(4):703–719, 2003.

Joshua E Auerbach and Josh C Bongard. Dynamic resolution in the co-evolution of morphology and control. In *Artificial Life XII: Proceedings of the Twelfth International Conference on the Synthesis and Simulation of Living Systems*, number EPFL-CONF-191277, pages 451–458. MIT Press, 2010a.

Nick Cheney, Robert MacCurdy, Jeff Clune, and Hod Lipson. Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. In *Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference*, pages 167–174. ACM, 2013.

John Rieffel, Davis Knox, Schuyler Smith, and Barry Trimmer. Growing and evolving soft robots. *Artificial life*, 20(1):143–162, 2014.

Joel Lehman and Kenneth O Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 211–218. ACM, 2011b.

David E Goldberg and John H Holland. Genetic algorithms and machine learning. *Machine learning*, 3(2):95–99, 1988.

Stefano Nolfi and Dario Floreano. Evolutionary robotics. the biology, intelligence, and technology of self-organizing machines. Technical report, MIT press, 2001.

Inman Harvey, Phil Husbands, Dave Cliff, Adrian Thompson, and Nick Jakobi. Evolutionary robotics: the sussex approach. *Robotics and autonomous systems*, 20(2): 205–224, 1997.

Stefano Nolfi, Dario Floreano, Orazio Miglino, and Francesco Mondada. How to evolve autonomous robots: Different approaches in evolutionary robotics. In *Artificial life IV: Proceedings of the 4th International Workshop on Artificial Life*, number LIS-CONF-1994-002, pages 190–197. MA: MIT Press, 1994.

Gillian M Hayes and John Demiris. *A robot controller using learning by imitation*. University of Edinburgh, Department of Artificial Intelligence, 1994.

Sridhar Mahadevan and Jonathan Connell. Automatic programming of behavior-based robots using reinforcement learning. *Artificial intelligence*, 55(2):311–365, 1992.

Nick Jakobi, Phil Husbands, and Inman Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *Advances in artificial life*, pages 704–720. Springer, 1995.

Jonathan D Hiller and Hod Lipson. Evolving amorphous robots. In *ALIFE*, pages 717–724, 2010.

Max Lungarella, Giorgio Metta, Rolf Pfeifer, and Giulio Sandini. Developmental robotics: a survey. *Connection Science*, 15(4):151–190, 2003.

Minoru Asada, Karl F MacDorman, Hiroshi Ishiguro, and Yasuo Kuniyoshi. Cognitive developmental robotics as a new paradigm for the design of humanoid robots. *Robotics and Autonomous Systems*, 37(2):185–193, 2001.

Juyang Weng. Developmental robotics: Theory and experiments. *International Journal of Humanoid Robotics*, 1(02):199–236, 2004.

Minoru Asada, Koh Hosoda, Yasuo Kuniyoshi, Hiroshi Ishiguro, Toshio Inui, Yuichiro Yoshikawa, Masaki Ogino, and Chisato Yoshida. Cognitive developmental robotics: a survey. *Autonomous Mental Development, IEEE Transactions on*, 1(1):12–34, 2009.

Maciej Komosiński and Adam Rotaru-Varga. Comparison of different genotype encodings for simulated three-dimensional agents. *Artificial Life*, 7(4):395–418, 2001.

Jimmy Secretan, Nicholas Beato, David B D Ambrosio, Adelein Rodriguez, Adam Campbell, and Kenneth O Stanley. Picbreeder: evolving pictures collaboratively online. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1759–1768. ACM, 2008.

Xin Yao and Yong Liu. A new evolutionary system for evolving artificial neural networks. *Neural Networks, IEEE Transactions on*, 8(3):694–713, 1997.

Kenneth Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

Kenneth O Stanley, David B D'Ambrosio, and Jason Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial life*, 15(2):185–212, 2009.

Joel Lehman and Kenneth O Stanley. Revising the evolutionary computation abstraction: minimal criteria novelty search. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 103–110. ACM, 2010.

Joel Lehman and Kenneth O Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *ALIFE*, pages 329–336, 2008.

Sebastian Risi, Sandy D Vanderbleek, Charles E Hughes, and Kenneth O Stanley. How novelty search escapes the deceptive trap of learning to learn. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 153–160. ACM, 2009.

Jean-Baptiste Mouret. Novelty-based multiobjectivization. In *New Horizons in Evolutionary Robotics*, pages 139–154. Springer, 2011.

Jean-Baptiste Mouret and Jeff Clune. An algorithm to create phenotype-fitness maps. In *Proc. of the Artificial Life Conf*, pages 593–594, 2012.

Deepak Trivedi, Christopher D Rahn, William M Kier, and Ian D Walker. Soft robotics: Biological inspiration, state of the art, and future research. *Applied Bionics and Biomechanics*, 5(3):99–117, 2008.

Rolf Pfeifer, Max Lungarella, and Fumiya Iida. The challenges ahead for bio-inspired'soft'robotics. *Communications of the ACM*, 55(11):76–87, 2012.

Filip Ilievski, Aaron D Mazzeo, Robert F Shepherd, Xin Chen, and George M Whitesides. Soft robotics for chemists. *Angewandte Chemie*, 123(8):1930–1935, 2011.

Robert F Shepherd, Filip Ilievski, Wonjae Choi, Stephen A Morin, Adam A Stokes, Aaron D Mazzeo, Xin Chen, Michael Wang, and George M Whitesides. Multigait soft robot. *Proceedings of the National Academy of Sciences*, 108(51):20400–20403, 2011.

Cecilia Laschi, Matteo Cianchetti, Barbara Mazzolai, Laura Margheri, Maurizio Follador, and Paolo Dario. Soft robot arm inspired by the octopus. *Advanced Robotics*, 26(7):709–727, 2012.

Sangok Seok, Cagdas Denizel Onal, Robert Wood, Daniela Rus, and Sangbae Kim. Peristaltic locomotion with antagonistic actuators in soft robotics. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 1228–1233. IEEE, 2010.

Shigeo Hirose and Yoji Umetani. The development of soft gripper for the versatile robot hand. *Mechanism and machine theory*, 13(3):351–359, 1978.

Jonathan D Hiller and Hod Lipson. Multi material topological optimization of structures and mechanisms. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1521–1528. ACM, 2009.

Siddharth Sanan, J Moidel, and CG Atkeson. A continuum approach to safe robots for physical human interaction. In *Int'l Symposium on Quality of Life Technology*, 2011.

Petros Faloutsos, Michiel Van De Panne, and Demetri Terzopoulos. Dynamic free-form deformations for animation synthesis. *Visualization and Computer Graphics, IEEE Transactions on*, 3(3):201–214, 1997.

Matthias Teschner, Bruno Heidelberger, Matthias Muller, and Markus Gross. A versatile and robust model for geometrically complex deformable solids. In *Computer Graphics International, 2004. Proceedings*, pages 312–319. IEEE, 2004.

Matthias Müller, Julie Dorsey, Leonard McMillan, Robert Jagnow, and Barbara Cutler. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 49–54. ACM, 2002.

Jonathan Hiller and Hod Lipson. Dynamic simulation of soft heterogeneous objects. *arXiv preprint arXiv:1212.2845*, 2012a.

Hod Lipson and Jordan B Pollack. Automatic design and manufacture of robotic lifeforms. *Nature*, 406(6799):974–978, 2000.

Jeff Clune, Benjamin E Beckmann, Charles Ofria, and Robert T Pennock. Evolving coordinated quadruped gaits with the hyperneat generative encoding. In *Evolutionary Computation, 2009. CEC'09. IEEE Congress on*, pages 2764–2771. IEEE, 2009.

Joshua E Auerbach and Josh C Bongard. Evolving cppns to grow three-dimensional physical structures. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pages 627–634. ACM, 2010b.

Jeff Clune and Hod Lipson. Evolving 3d objects with a generative encoding inspired by developmental biology. *ACM SIGEVOlution*, 5(4):2–12, 2011.

Michał Joachimczak and Borys Wróbel. Co-evolution of morphology and control of soft-bodied multicellular animats. In *Proceedings of the fourteenth international conference on Genetic and evolutionary computation conference*, pages 561–568. ACM, 2012.

Jonathan Hiller and Hod Lipson. Automatic design and manufacture of soft robots. *Robotics, IEEE Transactions on*, 28(2):457–466, 2012b.

Matthew Wall. Galib: A c++ library of genetic algorithm components. *Mechanical Engineering Department, Massachusetts Institute of Technology*, 87:54, 1996.

Jeffrey Antol, Philip Calhoun, John Flick, Gregory Hajos, Robert Kolacinski, David Minton, Rachel Owens, and Jennifer Parker. *Low cost mars surface exploration: The mars tumbleweed*. Citeseer, 2003.