Combining MAP-Elites and Incremental Evolution to Generate Gaits for a Mammalian Quadruped Robot

Jørgen Nordmoen, Kai Olav Ellefsen, and Kyrre Glette

Department of Informatics
University of Oslo
P.O. Box 1080 Blindern, 0316 Oslo, Norway
Email: jorgehn@ifi.uio.no

Abstract. Four-legged mammals are capable of showing a great variety of movement patterns, ranging from a simple walk to more complex movement such as trots and gallops. Imbuing this diversity to quadruped robots is of interest in order to improve both mobility and reach. Within the field of Evolutionary Robotics, Quality Diversity techniques have shown a remarkable ability to produce not only effective, but also highly diverse solutions. When applying this approach to four-legged robots an initial problem is to create viable movement patterns that do not fall. This difficulty stems from the challenging fitness gradient due to the mammalian morphology. In this paper we propose a solution to overcome this problem by implementing incremental evolution within the Quality Diversity framework. This allows us to evolve controllers that become more complex while at the same time utilizing the diversity produced by Quality Diversity. We show that our approach is able to generate high fitness solutions early in the search process, keep these solutions and perform a more open-ended search towards the end of evolution.

Keywords: map-elites, incremental evolution, quadruped, gait generation, movement primitives

1 Introduction

Legged robots have a high degree of mobility and can reach many places that are outside the reach of wheeled robots [1, 2]. This mobility enables legged robots to aid in many difficult situations and also work in environments optimized for human locomotion. However, designing a controller for a legged robot is a difficult challenge. Legged robots have many degrees of freedom and often require tight coordination to keep the body in balance. Manually designing such controllers is often time-consuming and as such *machine learning* is often seen as a promising alternative to generate the controller [3].

The field of Evolutionary Robotics (ER) takes inspiration from natural evolution to automatically create controllers for a large range of robots [4]. A recent

The final publication is available at Springer via https://doi.org/10.1007/978-3-319-77538-8_48

development in ER is to let the algorithms explore both diverse and high-quality solutions. This class of algorithms are called Quality Diversity (QD) [5, 6, 7, 8]. One interesting aspect of QD algorithms is their ability to produce a range of behaviors that can be utilized as a repertoire for subsequent selection [9, 10, 11]. These behavioral repertoires are especially interesting when utilized as a method to develop movement primitives that can later be operated by a higher-level controller [10]. By generating these primitives, higher-level abstractions can be created to facilitate re-use and sharing of controllers for different robots.

Four-legged, or *quadruped*, robots with legs underneath the body, pose a challenge for the control algorithm. Because the legs have to move out from under the robot, the center of gravity is shifted in such a way that the whole platform can become unstable [12, 13]. This instability makes it difficult to detect the gradient of the fitness function and often impedes the discovery of initial solutions. This is in contrast to other morphologies, such as the spider configuration in the Quadratot [14], where the legs are positioned on the side of the body.

The difficulty in discovering the initial solutions is described as the *bootstrap problem* in ER [4, 15, 16]. The bootstrap problem occurs when the initial population is difficult to generate and is of poor quality. In complex search spaces this often occurs because initial individuals are not able to complete the task. In the context of quadruped gait generation, the bootstrap problem occurs because any gait must have a high degree of coordination between all four legs. With many degrees of freedom in each leg, coordination becomes increasingly unlikely requiring several beneficial changes to occur together for the motion to become synchronized. Since many ER algorithms typically initialize the search randomly, the problem becomes even greater, because evolution might use many evaluations in unproductive regions of the search space [15].

In this paper we propose to combine Multi-dimensional Archive of Phenotypic Elites (MAP-Elites) [7, 8] with incremental evolution. Our contribution is twofold, firstly we show that it is possible to use MAP-Elites to generate a diverse set of high-quality movement primitives for the difficult quadruped morphology. Secondly, we study the effect of incremental controller complexification within the MAP-Elites framework.

2 Background

This section will review background material on gait generation and incremental evolution before describing QD. A special focus will be given to MAP-Elites, one of the more popular QD algorithms, as well as the basis for our algorithm.

2.1 Gait Generation

Generating gaits has a long history within machine learning and ER. Some early examples of generating gaits can be found in [1, 17, 18]. A very successful example of gait generation in ER is the work of [3] which used an Evolutionary Algorithm

(EA) to optimize the gait of the Sony AIBO robot, the result of which ended up in the commercial release of the robot.

In recent years a large body of research has focused on the control structure, and learning of such structures, in order to produce a gait. One such approach is using Artificial Neural Networks (ANNs) to produce the control signals for the legs [19]. By using EAs, optimizing network weights, parameters and even whole structures has been shown effective at creating controllers for legged robots [20, 21].

As hardware improves, real-world evolution–evaluating the fitness of a real robot instead of a simulated version–has also been applied to the problem of gait generation. This approach avoids the reality gap problem and can produce good results [14, 22]. However, this approach is difficult to apply to mammalian morphologies and open-ended evolution of the control structure has not been demonstrated [23].

Since its introduction, QD has been used extensively to generate gaits. One example is [24] which compared different control structures within the QD framework, they showed that QD can be used with many different control structures, but noted that the effectiveness of QD can depend on the control structure.

2.2 Incremental Evolution

The bootstrap problem is one of the fundamental problems within ER [16, 25]. The problem arises when the fitness function is not able to guide the set of randomly generated initial solutions to improve [15]. One solution to the bootstrap problem is incremental evolution [15]. Incremental evolution starts by decomposing the goal into smaller sub-tasks that evolution is able to solve individually. When the desired fitness is reached in a sub-task, the task is either updated to the next sub-task or the task is made more complex. To benefit from the previous solution special care is taken when the task is decomposed so that the previous solution is a component to solve the next sub-task. Once all sub-tasks have been solved the tasks can be combined into the goal task. Alternatively, if controller complexification is used the problem can be considered solved when the controller is complex enough to solve the goal.

In their seminal paper on incremental evolution Gomez [15] used task decomposition to evolve a neural network in a predator-prey scenario. Similar task decomposition has been used to evolve gait and navigation controller for a 6-legged robot [26], evolving control for soccer players [27] and cooperative phototaxis with hole avoidance [28]. Other forms of incremental evolution has also been show to be effective. Bongard [29] showed that changing the morphology over both the 'lifetime' and during evolution increases performance and robustness of the robot's controller. Bongard [30] also demonstrated that combining environmental and morphological incremental evolution is not only possible, but yields better performance than either version alone. Mouret [31] identified some of the challenges with incremental evolution, such as the difficulty in specifying how a task should be decomposed. Their innovative solution of task decomposition with a multi-objective EA was able to overcome some of these problems,

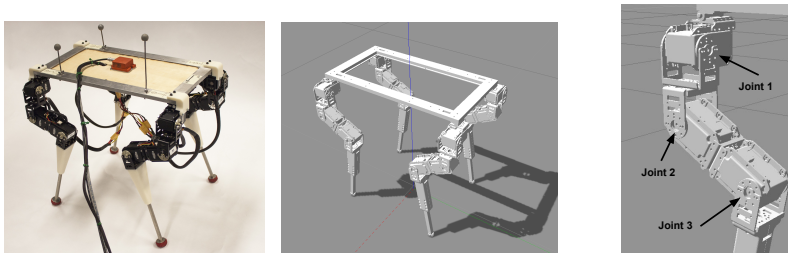
however, incremental evolution still requires careful consideration in its implementation.

2.3 Quality Diversity

A recent advance in ER is the use of QD algorithms to overcome some of the fundamental problems within the field. This new class of algorithms eschews some of the traditional focus on fitness alone and instead concentrates on the behavior of the system [5, 32]. These algorithms have shown a remarkable ability to solve problems that before were considered too complex [6, 11, 33].

QD algorithms like MAP-Elites [8] separates the behavior, of the system in question, from the fitness. This separation allows QD algorithms to search for interesting differences in behavior rather than solely focusing on fitness. The separation of behavior and fitness makes QD different from a Multi-Objective Evolutionary Algorithm (MOEA) since lower performing solutions are kept as long as they are behaviorally different from what has already been encountered. That is, even though a solution is not on the Pareto front it could still be included if it is behaviorally different from other solutions. The advantage of this diversity is that the search is less likely to be stuck in local optima, and for high performing solutions to be built on top of similar, lower performing solutions [7].

A recent development within QD [10] showed that it was possible to create control primitives by pairing MAP-Elites with an ANN to create control abstractions for a simulated vehicle. This work built on the strengths of QD to discover a diverse set of high quality control primitives that could be paired with higher-level control abstractions. The ANN was used to select control primitives and was shown to be able to transition between different vehicle control modes enabling the authors to abstract the underlying vehicle dynamics from the high-level ANN control system. The work demonstrated the strength of using these higher-level abstractions in creating complex controllers for unknown robots possibly enabling new robots to be more easily developed and tested.



(a) Real-world (b) Simulated (c) Joint configuration

Fig. 1: The quadruped robot used in this paper.

3 Approach

This paper combines MAP-Elites with incremental complexification to evolve a set of controllers for a four-legged quadruped robot. The robot (Fig. 1) was developed as an experimental platform to perform real-world evolution *in hardware* [23]. Each leg has three degrees of freedom as shown in Fig 1c, for a total of twelve degrees of freedom, with each joint controlled by an internal PID-controller. In order to run our experiments the robot was simulated using Gazebo version 7.8.1[1] with ODE[2] in conjunction with Robot Operating System (ROS)[3]. In addition, we used the *SFERESv2* framework [34] with its default MAP-Elites implementation[4]. To understand how incremental evolution is integrated with MAP-Elites we will introduce the gait controller and explain how this controller can be complexified.

3.1 Gait Controller

To create a 'complexifying', or 'upgrading', gait controller we started by parameterizing a simple quadruped walk, where the parameters are *amplitude*, *phase*, *duty cycle*, *offset* and *gait period* for each joint. These parameters describe a continuous first-order spline which represents the commanded angle of a joint[5]. Complexification is performed by locking, not allowing evolution to mutate, selected parameters to default values and later unlocking these same parameters, this is illustrated in Fig. 2 for a simple four valued genome. The parameterization makes it trivial to do non-destructive upgrades, retaining the phenotypic expression, which is important when integrating with MAP-Elites since individuals in the population are not re-evaluated. In the experiments all individuals in the population are upgraded at the same time to ensure that mutations are allowed to happen for the newly unlocked parameters. The parameters used for the different experiments are explained in Section 3.3.

3.2 Evolutionary Setup

The shared MAP-Elites parameters can be found in Table 1. Each individual is simulated for 10 seconds and an evaluation ends if the individual falls over. The fitness of each individual is

$$fitness(n) = \begin{cases} T_i & \text{if the robot fell over} \\ T_i + stability(n) & \text{otherwise} \end{cases} \quad (1)$$

[1] `http://gazebosim.org/`
[2] `http://www.ode.org/`
[3] `http://ros.org`
[4] Source code: `https://folk.uio.no/jorgehn/dyret_map_gaits-0.1.0.zip`.
[5] For further details see the source code.

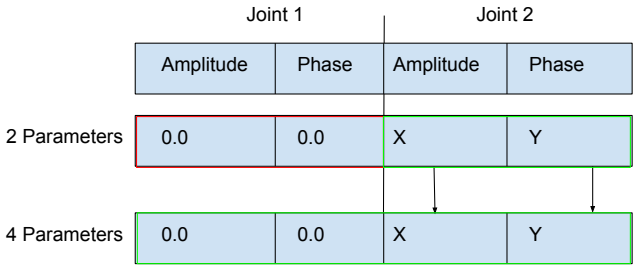


Fig. 2: Parameter upgrade for two simple joints. The full configuration is parameterized by four values, two values for each joint. In the first scenario two of the values are locked, marked in red, and two are unlocked. To upgrade the configuration, we copy the two values, X and Y, and unlock the two remaining parameters. The figure illustrate how two different configurations can behave identically, but the '4 Parameter' configuration has the possibility to exhibit new behavior as it can actuate 'Joint 1'.

where T_i is the time the individual was upright, and *stability* is defined as

$$stability(n) = \begin{cases} C - SM(\omega_x, \omega_y) & \text{if } SM(\omega_x, \omega_y) < C \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where SM is the Squared Magnitude of the x and y components of the body angular momentum over the evaluation period and C is a constant allowing for maximization of SM. This fitness function ensures that MAP-Elites is able to progress even if none of the individuals are able to walk the full evaluation time. By using SM of the angular momentum MAP-Elites optimizes the stability in each grid cell, where less angular momentum is interpreted as a more stable gait. This was chosen to increase viability of the gait and matches the MAP-Elites notion of optimizing fitness while letting behavior characteristics explore the behavior space.

The behavior characteristics used in the simulation are inspired by [10] and are designed to support higher-level control abstractions. The behavior characteristics are average turn rate defined as

$$\frac{1}{N}\sum_{i=2}^{N}\frac{(\psi_i - \psi_{i-1})}{(t_i - t_{i-1})} \tag{3}$$

and average velocity defined as

$$\frac{1}{N}\sum_{i=1}^{N}\overline{v}_i \tag{4}$$

for x and y dimensions respectively, where N is the number of samples *before* the robot fell, ψ is the yaw in radians, t_i is the time of sample i and $\overline{v}$ is the

velocity of the robot. The behavior characteristics drive the search to explore a variety of velocities and turn rates while fitness optimizes for stability of each behavior.

Evaluations	*30 000* Generations: 300 Batch size: 100
Initial population	1000
Evaluation time	10 seconds
Recombination	None
Mutation	Type: Gaussian σ: 0.2 Probability: 1.0
Behavior characteristics	Dimensions: 2 X-axis: turn rate Y-axis: Average speed

Table 1: MAP-Elites simulation parameters.

3.3 Experimental Setup

To investigate if incremental evolution can supplement MAP-Elites we tested four different configurations of the gait controller. For the experiments in this paper, *phase*, *duty cycle* and *offset* are kept static in all configurations. For each leg the phase parameter is kept identical for all joints set to 0.00, 0.75, 0.50 and 0.25 for the front left, front right, back left and back right leg respectively where the value is a percentage of the total gait period. The duty cycle is set to 0.25 and the offset for each joint is set to the resting pose of the robot. This increases the probability of discovering viable gaits while still allowing enough freedom to differentiate the four configurations. The four configurations are described below. The setup consists of three base-configurations which function as reference implementation and are compared to our incremental configuration.

Simple To ensure that the gait controller is capable of producing viable gaits the first configuration tested restricts almost all parameters of the gait. In this configuration most parameters are set to best practices and only a few parameters are evolved. The evolved parameters are as follows, see Figure 1c,

- First parameter describes amplitude of Joint 1.
- Second parameter describes amplitude of Joint 2.
- Third parameter describes amplitude of Joint 3.

These parameters are then replicated for each leg and movement is ensured by different phases between the legs.

Medium The next configuration tested relaxes a few more restrictions and allows the amplitude of each joint in the left and right legs to be evolved separately. The parameters evolved are

- The first three (1-3) parameters describe the amplitude of the joints in the two left legs.
- The next three (4-6) parameters describe the amplitude of the joints in the two right legs.

This configuration has the potential to explore more gaits along the X axis compared to the *simple* configuration, due to the decoupling between the left and right side of the robot. Since the left and right side can have independent amplitudes, the gait pattern has the potential to create behaviors more suited to turning. However, the increase in number of parameters also requires more coordination.

Complex In this configuration each leg has independent amplitude control of all joints, giving 12 parameters to optimize. This configuration is the least restrictive of the base-configurations, which could lead to more diverse gaits with better performance. The expressiveness could also be a hindrance as the number of parameters that must be coordinated is larger.

Incremental Controller The incremental controller is a combination of the above configurations. During evolution this configuration will start with the 'Simple' configuration and will change, first, to the 'Medium' and then lastly to the 'Complex' configuration. This controller tests if incremental evolution can be combined with MAP-Elites to produce diverse, high performing solutions. The gait is incrementally upgraded at static points during the evolution, fixed to $\frac{1}{3}$ of the total evaluations for each sub-configuration.

In the results we have also added a configuration called *Incremental 2* which also performs incremental evolution, but instead of starting with 'Simple' starts in the 'Medium' configuration and *Incremental 3* which starts with 'Simple' and upgrades to 'Complex'. Both of these configurations incrementally complexify after $\frac{1}{2}$ of the total evaluations have been performed. These configurations are included to gain insight into the effect of upgrading and how upgrading is affected by the initial configuration.

4 Experimental results

To compare the different configurations and to understand how incremental evolution performs within the MAP-Elites framework we ran each configuration with 15 repetitions and used the metrics described in [7] to analyze the results. These metrics, *reliability*, the average fitness for all cells in the map, *coverage*, the percentage of filled cells, and *precision*, the average fitness of filled cells, give an overview of the state in the map. Since we are evolving gait primitives

we are interested in both a diverse set of solutions (large coverage) and high performance (high precision). Reliability then becomes the product of these two objectives and gives a summary of the performance.

In the results we annotate the three incremental configurations, described in Section 3.3, as 'Incr', 'Incr 2' and 'Incr 3' respectively, see Fig. 4. Note also that even though fitness is evaluated even if the solution fell these solutions are not included in the results presented below, the individuals are removed before any calculations take place. In other words, the results only considers gait controllers that were able to walk the full simulation time without falling over. For the results below all fitness values have been normalized by subtracting T_i and dividing by the constant C.

To further illustrate the results, we have plotted the resulting map, in Fig. 3, from a few select generations for representative runs of each configuration. These maps give a good overview of the distinction between precision and coverage. From the maps it is clear to see that 'Incremental' and 'Simple' find higher performing solutions while 'Complex' is better able to explore the search space.

4.1 Precision

Fig. 4 shows the average fitness for the filled cells for each of the configurations tested throughout evolution. For the static configurations the precision slowly increases over the generations. Most improvement is seen in the 'Complex' configuration while 'Simple' and 'Medium' start higher and have little to no increase over the course of evolution. For the incremental configurations the plots look quite different and actually decrease towards the end of the evolutionary run. The large difference in precision seen for the static configurations is likely due to the relative difference in difficulty of finding good solutions. For the 'Simple' configuration the parameters require very little coordination which seems to result in higher precision. This is also very evident from the initial random population where the difference in precision should be directly correlated to the difficulty in randomly generating a functioning gait. Analyzing the box plot for generation 300, in Fig. 4 top right, the results show very little spread and most of the variation is in the 'Complex' configuration.

4.2 Coverage

Fig. 4, middle row, shows the number of filled cells over all generations and a box plot of the last generation. From the generational plot it is clear that the 'Complex' configuration is best able to explore the search space. The two other static configurations achieve significantly less coverage, which is expected considering the limitations in the number of parameters and resulting gaits. For the incremental configurations we clearly see that more parameters open up more possibilities as each configuration quickly explores more of the search space after each upgrade, generation $100, 200$ for 'Incremental' and 150 for the two-other incremental configurations.

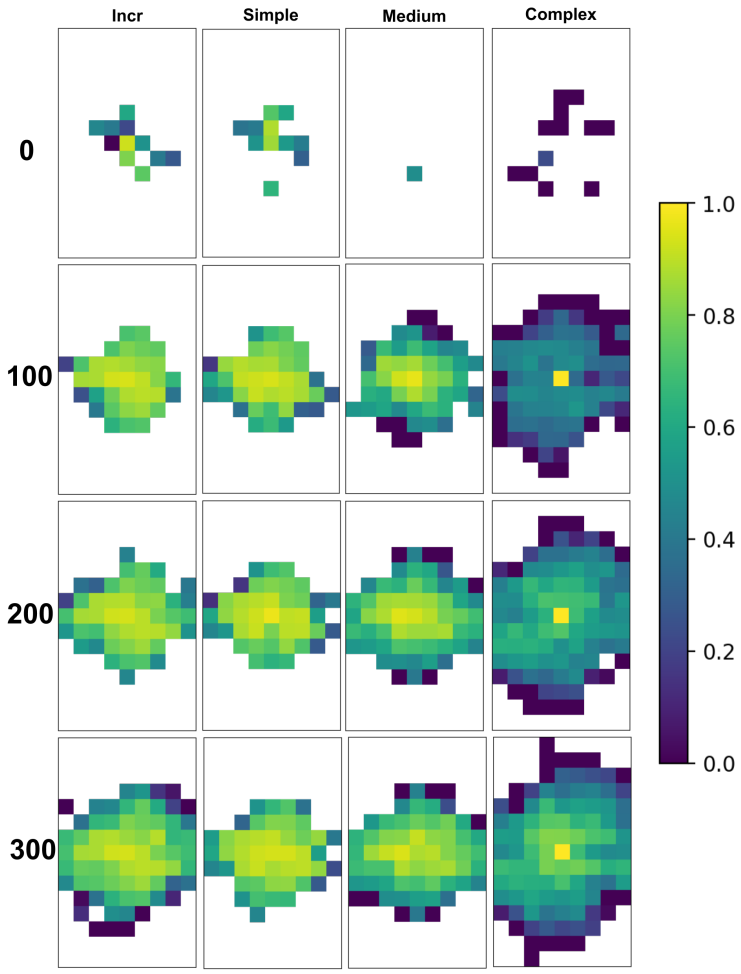


Fig. 3: Illustration of evolved maps for generation, 0, 100, 200 and 300. The X and Y axis represent the behavior dimensions while the color represents fitness where bright yellow is higher fitness. For the 'Incremental' configuration the upgrade happens at generation 100 and 200. The figure illustrates the distinction between precision and coverage, where 'Complex' spans a much wider area with lower performing solutions while 'Simple' has fewer solutions with higher fitness.

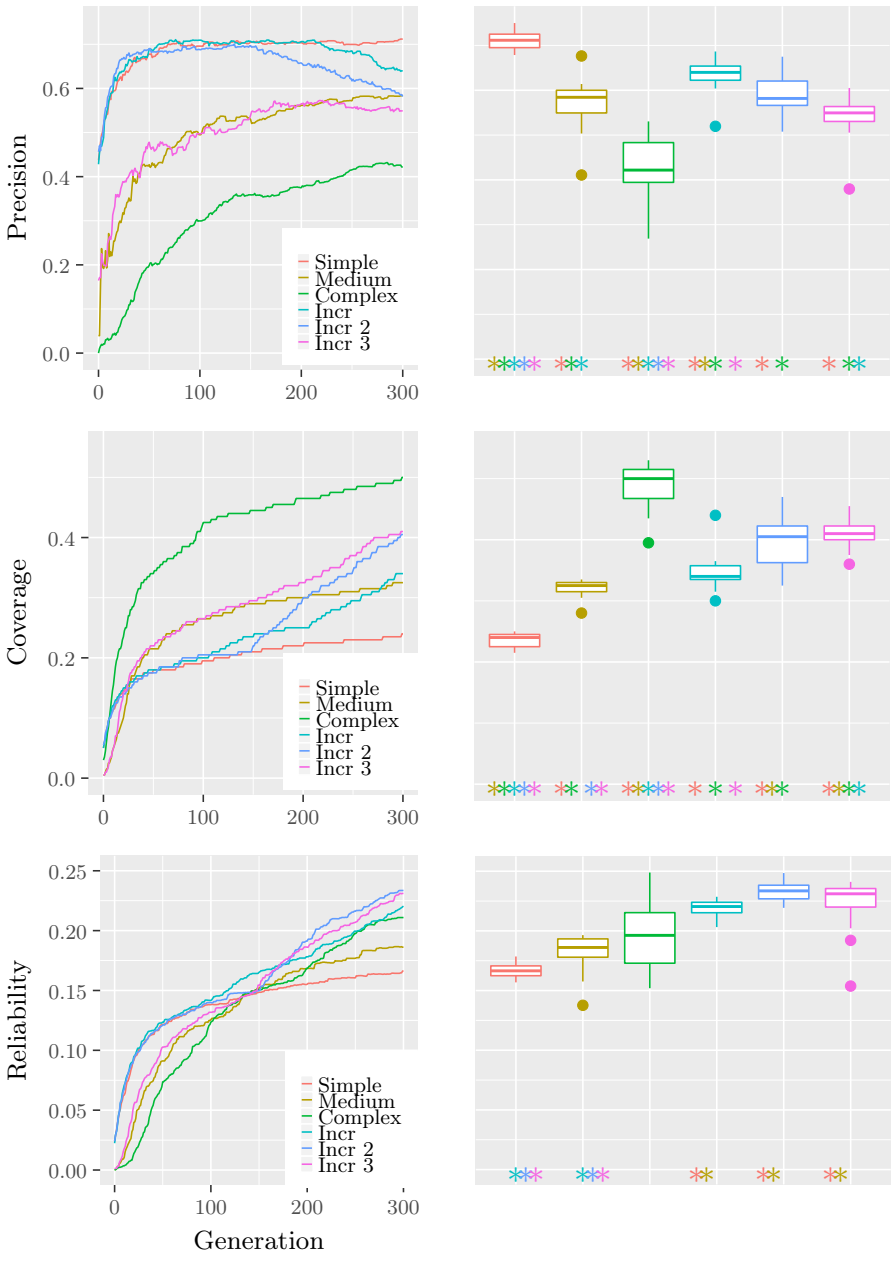


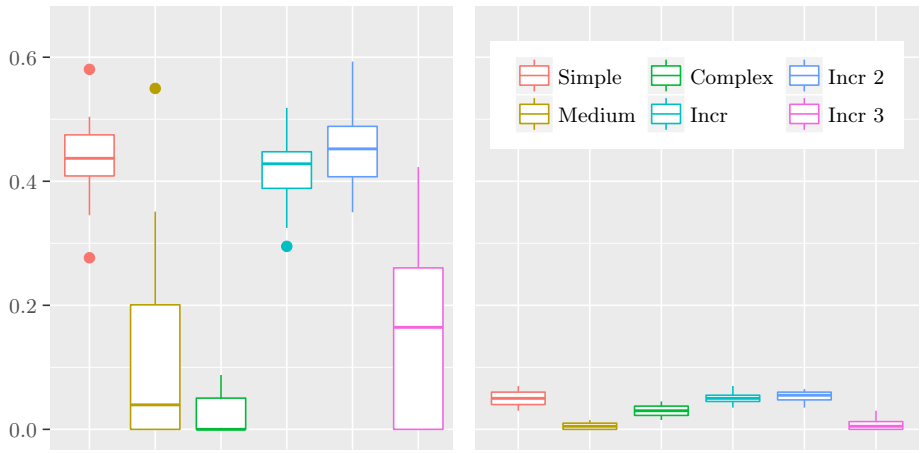
Fig. 4: The plots show precision, coverage and reliability. The median is plotted over all generations, left, and a box plot is shown for the last generation on the right. The box plots also show the result of a pairwise Wilcoxon Rank Sum test, adjusted using Holm's method, where an *asterisk* corresponds to a significant difference at the $p < 0.001$ level.

It is also interesting to note that the initial coverage for all configurations is about the same, but the coverage quickly diverges in subsequent generations. The box plots also show that the difference is significant between several of the configurations for the final generation.

4.3 Reliability

To get an overall impression of the performance of the tested configurations we can combine precision and coverage into reliability as seen in Fig. 4. Reliability is defined as the average fitness of all cells divided by the number of cells, empty cells are given a fitness of zero. From the generational figure it seems that 'Complex' performs better than the two other static configurations although the difference is not statistically significant. The three incremental configurations show different growths throughout evolution, but by the end of the run start to converge to the same performance.

All of the incremental configurations show large increases in reliability after upgrading, which is correlated with the increase in coverage, as noted in the previous section. These configurations also have higher reliability throughout the beginning of the evolution compared to the 'Complex' configuration correlated with the large difference in precision.



(a) Precision for the initial population. (b) Coverage for the initial population

Fig. 5: These figures show precision and coverage for the initial population. The figures illustrate the difficulty in discovering the initial population depending on the 'freedom' of the controller. This can be seen as coverage is essentially equal for all configurations, yet the fitness of the initial populations are very different, as illustrated by the difference in precision.

5 Discussion

The main hypothesis explored in this paper is that MAP-Elites combined with incremental evolution can be used for the difficult task of generating gait primitives for a mammalian quadruped robot. The results show that the quality and quantity of the primitives varies between the different static configurations. In contrast the incremental approach is able to develop a large repertoire of high quality solutions that span a larger area in the search space. It can also be observed that the variance for all three incremental approaches are much lower than for the best static configuration, as seen in Fig. 4, this could indicate that the incremental configurations are more consistently finding diverse and high fitness solutions. From manual inspection of the gaits produced, all configurations are able to discover satisfying controllers that exhibit desired behavior[6].

From the precision plot in Fig. 4, it can be seen that complexity is related to fitness, as more complex configurations achieve lower precision scores. The lower precision can be attributed to the difficulty in coordinating the joints. To further explore this we plot the initial population in Fig. 5. Even though the difference in coverage is low the difference in precision is large.

Because of the limitations imposed on the 'Simple' and 'Medium' configurations they are not able to gain the same coverage as the 'Complex' configuration. This can be attributed to the forced coordination imposed on these configurations. These results are interesting because they illustrate the difficulty in designing an algorithm to overcome the bootstrap problem. By restricting the configurations they are able to more quickly discover high performing solutions, as evident in their precision Fig. 4, but are not able to produce the same range of behaviors as the 'Complex' configuration. We can see that in the reliability that the incremental configurations seem to be able to get the benefit of both–however, because of the limited number of evaluations in this paper–they are not able to completely surpass the 'Complex' configuration.

The incremental approach is able to generate higher fitness controllers as well as explore more of the search space, as seen in the results Fig. 4. This is because it is able to keep the high precision from before the upgrade, and also explore the new behavior space after the upgrade. The improvement is evident in each of the three incremental approaches and is a result of the increase in coverage.

6 Conclusion and Future Work

We have investigated how incremental evolution can be combined with MAP-Elites to create movement primitives for quadruped robots. In the experiments we compared several gait parameterizations of different levels of complexity with a controller that is able to incrementally complexify. The results show that MAP-Elites was able to create viable controllers, and–when combined with incremental evolution–also produced a diverse set of solutions with high fitness. This indicates

[6] For videos see: `https://folk.uio.no/jorgehn/map_gaits/`

that for more open-ended controllers incremental complexification could be a promising approach for introducing some guidance into the search.

This paper also shows that evolving movement primitives for mammalian quadruped robots is possible and a large repertoire of gait patterns can be evolved. By showing that movement primitives are achievable with such a difficult morphology, we open the possibility to apply the technique to a large group of robots so far not explored with Quality Diversity techniques.

A natural next step for this research is to verify that the evolved gait primitives are able to function in the real-world. An extension to this work could also explore performing the complexification adaptively as a means to achieve faster convergence.

Acknowledgments. Supported by The Research Council of Norway as a part of the Engineering Predictability with Embodied Cognition (EPEC) project, under grant agreement 240862.

Bibliography

[1] Wettergreen, D., Thorpe, C.: Gait generation for legged robots. In: IEEE International Conference on Intelligent Robots and Systems (1992)

[2] Bares, J.E., Whittaker, W.L.: Configuration of autonomous walkers for extreme terrain. The International Journal of Robotics Research 12(6), 535–559 (1993)

[3] Hornby, G.S., Takamura, S., Yokono, J., Hanagata, O., Yamamoto, T., Fujita, M.: Evolving robust gaits with AIBO. In: Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on. vol. 3, pp. 3040–3045. IEEE (2000)

[4] Doncieux, S., Bredeche, N., Mouret, J.B., Eiben, A.E.G.: Evolutionary robotics: what, why, and where to. Frontiers in Robotics and AI 2, 4 (2015)

[5] Pugh, J.K., Soros, L.B., Stanley, K.O.: Quality diversity: A new frontier for evolutionary computation. Frontiers in Robotics and AI 3, 40 (2016)

[6] Lehman, J., Stanley, K.O.: Exploiting open-endedness to solve problems through the search for novelty. In: ALIFE. pp. 329–336 (2008)

[7] Mouret, J.B., Clune, J.: Illuminating search spaces by mapping elites. arXiv preprint arXiv:1504.04909 (2015)

[8] Cully, A., Clune, J., Tarapore, D., Mouret, J.B.: Robots that can adapt like animals. Nature 521(7553), 503–507 (2015)

[9] Cully, A., Mouret, J.B.: Behavioral repertoire learning in robotics. In: Proceedings of the 15th annual conference on Genetic and evolutionary computation. pp. 175–182. ACM (2013)

[10] Duarte, M., Gomes, J., Oliveira, S.M., Christensen, A.L.: EvoRBC: Evolutionary repertoire-based control for robots with arbitrary locomotion complexity. In: Proceedings of the 18th annual conference on Genetic and evolutionary computation. ACM (2016)

[11] Cully, A., Mouret, J.B.: Evolving a behavioral repertoire for a walking robot. Evolutionary computation 24(1), 59–88 (2016)
[12] Van de Panne, M., Lamouret, A.: Guided optimization for balanced locomotion. In: Computer Animation and Simulation'95, pp. 165–177. Springer (1995)
[13] de Santos, P.G., Garcia, E., Estremera, J.: Quadrupedal locomotion: an introduction to the control of four-legged robots. Springer Science & Business Media (2007)
[14] Yosinski, J., Clune, J., Hidalgo, D., Nguyen, S., Zagal, J., Lipson, H.: Evolving robot gaits in hardware: the HyperNEAT generative encoding vs. parameter optimization. In: Proceedings of the 20th European Conference on Artificial Life. pp. 890–897 (2011)
[15] Gomez, F., Miikkulainen, R.: Incremental evolution of complex general behavior. Adaptive Behavior 5(3-4), 317–342 (1997)
[16] Silva, F., Duarte, M., Correia, L., Oliveira, S.M., Christensen, A.L.: Open issues in evolutionary robotics. Evolutionary Computation 24(2), 205–236 (2016)
[17] Brooks, R.A.: A robot that walks; emergent behaviors from a carefully evolved network. Neural computation 1(2), 253–262 (1989)
[18] Matarić, M., Cliff, D.: Challenges in evolving controllers for physical robots. Robotics and autonomous systems 19(1), 67–83 (1996)
[19] Billard, A., Ijspeert, A.J.: Biologically inspired neural controllers for motor control in a quadruped robot. In: Neural Networks, 2000. IJCNN 2000, Proceedings of the IEEE-INNS-ENNS International Joint Conference on. vol. 6, pp. 637–641. IEEE (2000)
[20] Clune, J., Beckmann, B.E., Ofria, C., Pennock, R.T.: Evolving coordinated quadruped gaits with the HyperNEAT generative encoding. In: 2009 IEEE Congress on Evolutionary Computation. pp. 2764–2771. IEEE (2009)
[21] Lee, S., Yosinski, J., Glette, K., Lipson, H., Clune, J.: Evolving gaits for physical robots with the HyperNEAT generative encoding: The benefits of simulation. In: European Conference on the Applications of Evolutionary Computation. pp. 540–549. Springer (2013)
[22] Zykov, V., Bongard, J., Lipson, H.: Evolving dynamic gaits on a physical robot. In: Proceedings of Genetic and Evolutionary Computation Conference, Late Breaking Paper, GECCO. vol. 4 (2004)
[23] Nygaard, T.F., Tørresen, J., Glette, K.: Multi-objective evolution of fast and stable gaits on a physical quadruped robotic platform. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI) (2016)
[24] Tarapore, D., Clune, J., Cully, A., Mouret, J.B.: How do different encodings influence the performance of the MAP-Elites algorithm? In: Genetic and Evolutionary Computation Conference (2016)
[25] Mouret, J.B., Doncieux, S.: Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity. In: 2009 IEEE Congress on Evolutionary Computation. pp. 1161–1168. IEEE (2009)
[26] Filliat, D., Kodjabachian, J., Meyer, J.A., et al.: Incremental evolution of neural controllers for navigation in a 6-legged robot. In: Proceedings of the

Fourth International Symposium on Artificial Life and Robots. pp. 753–760 (1999)
[27] Whiteson, S., Kohl, N., Miikkulainen, R., Stone, P.: Evolving soccer keepaway players through task decomposition. Machine Learning 59(1), 5–30 (2005)
[28] Christensen, A.L., Dorigo, M.: Incremental evolution of robot controllers for a highly integrated task. In: International Conference on Simulation of Adaptive Behavior. pp. 473–484. Springer (2006)
[29] Bongard, J.: Morphological change in machines accelerates the evolution of robust behavior. Proceedings of the National Academy of Sciences 108(4), 1234–1239 (2011)
[30] Bongard, J.: Morphological and environmental scaffolding synergize when evolving robot controllers: artificial life/robotics/evolvable hardware. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation. pp. 179–186. ACM (2011)
[31] Mouret, J.B., Doncieux, S.: Incremental evolution of animats' behaviors as a multi-objective optimization. From Animals to Animats 10 pp. 210–219 (2008)
[32] Auerbach, J.E., Iacca, G., Floreano, D.: Gaining insight into quality diversity. In: Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion. pp. 1061–1064. ACM (2016)
[33] Lehman, J., Stanley, K.O.: Evolving a diversity of virtual creatures through novelty search and local competition. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation. pp. 211–218. ACM (2011)
[34] Mouret, J.B., Doncieux, S.: SFERESv2: Evolvin' in the multi-core world. In: Proc. of Congress on Evolutionary Computation (CEC). pp. 4079–4086 (2010)