Effects of compliant and structural parts in evolved modular robots

Emma Stensby Norstein[1*], Frank Veenstra[1,2], Kai Olav Ellefsen[1], Tønnes Nygaard[1,3] and Kyrre Glette[1,2]

[1]Department of Informatics, University of Oslo, Oslo
[2]RITMO, University of Oslo, Oslo
[3]Department of Mechanical, Electronic and Chemical Engineering, Oslo Metropolitan University, Oslo
[*]emmaste@ifi.uio.no

Abstract

A striking difference between animals and traditional robots is that the latter usually have rigid and non-flexible bodies. Animals, on the other hand, exhibit highly adapted traits, such as elastic tendons. The tendons work as springs, storing and releasing kinetic energy during an animal's gait cycle. Springs have been used in some hand designed robots for similar benefits. However, little research has been done on springs in robots with evolving morphology. We examine the use of compliant and structural modules in modular robots, using a standard evolutionary algorithm. We also look at connections between spring stiffness and robot size using the quality diversity algorithm MAP-Elites. We found that the modular robots evolved to use elastic actuators, and that structural modules enabled morphologies that use less actuators, but still achieve the same walking speed as the robots with actuators in every module. We also observe some indications that larger robots may require lower elasticity.

Introduction

Complex animals have emerged from millions of years of evolutionary pressure. These animals exhibit many adaptive traits that increase their efficiency in various tasks. One such trait is the reuse of potential and kinetic energy during locomotion. Elastic tendons can store and release kinetic energy, which results in a minimization of work required for a limb (Alexander, 1984). For example, a vertebrate tendon's stress-strain curve shows that up to 93% of kinetic energy can be recovered when a tendon is loaded and unloaded (Biewener, 2003). Compliant parts have been used in hand designed robots to provide similar benefits to that of a tendon in an animal (Zhou and Bi, 2012). Since the purpose of tendons is to minimize work through recovering kinetic energy, and because we see a selection pressure towards elastic tendons in nature, this paper tries to evaluate whether there is a selective pressure towards springiness when evolving modular robots.

Evolutionary robotics (Doncieux et al., 2011) attempts to harness the adaptive power of evolution, and use it as a tool to design the morphology or controller of machines (Lipson and Pollack, 2000; Cully et al., 2015). Evolution of virtual or mechanical robots can also be used to gain insights into evolutionary processes, and to investigate biological questions (Roberts et al., 2014). Although virtual creatures are merely an abstraction of the real world, the ability to rapidly process many generations by means of parallel computation is a benefit, allowing experiments on their evolution to be carried out in hours or days. Furthermore, one can easily change experimental conditions, such as morphological building blocks, encodings, or environmental properties (Sims, 1994).

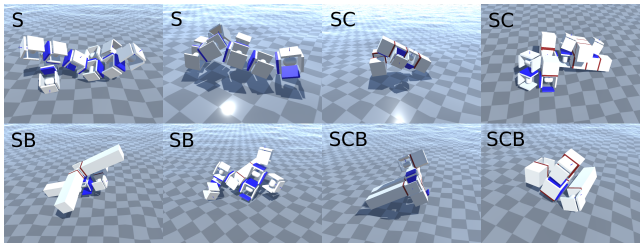

Figure 1: Examples of robots evolved with the standard evolutionary algorithm. The letter codes refer to the modules available for the algorithm. S - serial elastic actuator, C - connector block, B - beam.

Modular robotics concerns robots with adaptable bodies, that can be modified through assembly and reconfiguration of modular components (Stoy et al., 2010; Yim et al., 2007). As the morphology of modular robots can easily be modified, and thus be adapted through evolution, these robots can be used to investigate evolutionary pressures on morphology. Many different modular robot systems have been proposed. They range from robots made from building blocks such as actuators, connectors and structural elements (Zhao et al., 2020; Auerbach et al., 2014), to soft robots made of voxels in a lattice structure, where the voxels simulate the properties of different types of tissue (Cheney et al., 2014), and even to robots made from live cells (Blackiston et al., 2021). Although there are many different approaches to evolving robot morphology, there has been little research on how different module designs and properties are utilized by the evolutionary algorithms. Some research has been done

on the number of connection sites on a module (Liu et al., 2017), and on the length of a module (Moreno and Faina, 2020). However, both of these studies used exclusively actuated joint modules in the limbs of the robots. To further explore the usage of different types of modules, we will explore the selection pressures, in a standard evolutionary algorithm, towards structural elements and a serial elastic actuator with evolvable elasticity.

Just like for animals, elastic parts can be useful in robotics (Zhou and Bi, 2012). A series elastic actuator (Pratt and Williamson, 1995) is an actuator, such as a servo motor, with a spring attached along the actuated axis of the motor. This type of actuator is common in hand designed robots (Pratt and Krupp, 2004). A serial elastic actuator can provide many benefits, such as reduction of stress on joints and increased energy efficiency. It has also been used in robots that work with or alongside humans to increase safety in human-robot interaction (Yu et al., 2015). Serial elastic actuators have even been used in modular robots previously, although they were only tested with handcrafted morphologies (Kalouche et al., 2015).

In addition to exploring the selection pressure towards springiness in modular robots, we also want to explore how springiness is connected to morphology. In nature larger animals tend to have stiffer leg springs (Farley et al., 1993). We want to explore if the same connections are applicable in the context of modular robotics. That is, will larger modular robots evolve to have stiffer spring than smaller ones?

While there have been studies on the effects of selection pressures and environments on evolved robot bodies and brains (Auerbach and Bongard, 2014; Miras et al., 2018b; Miras and Eiben, 2019), these relations have been surprisingly difficult to demonstrate (Miras and Eiben, 2019; Miras and Ferrante, 2020). The reasons for this remain convoluted, but design factors such as developmental encodings and controllers can bias the search space (Miras et al., 2018a; Veenstra and Glette, 2020). Moreover, the search landscape, in particular for modular robots, can be complex and lead to premature convergence challenges (Faíña et al., 2013; Cheney et al., 2016).

To explore the connections between evolved springiness and size we will need to evolve robots with a variety of sizes. A standard evolutionary algorithm that converges to one solution is not capable of this on its own, so we opt to use a quality-diversity algorithm (Chatzilygeroudis et al., 2021) for this purpose. Quality-diversity algorithms have been used in modular robotics to increase diversity and avoid converging prematurely to a local optima (Nordmoen et al., 2021). The quality-diversity algorithm MAP-Elites (Mouret and Clune, 2015) will be especially useful to explore the connections between morphology and springiness. In MAP-Elites individuals with different properties are arranged in a grid, and an elite is found within each cell of the grid. By creating a grid with robot springiness along one axis, and robot size along the other, we can analyse the benefit of different amounts of springiness across different robot sizes.

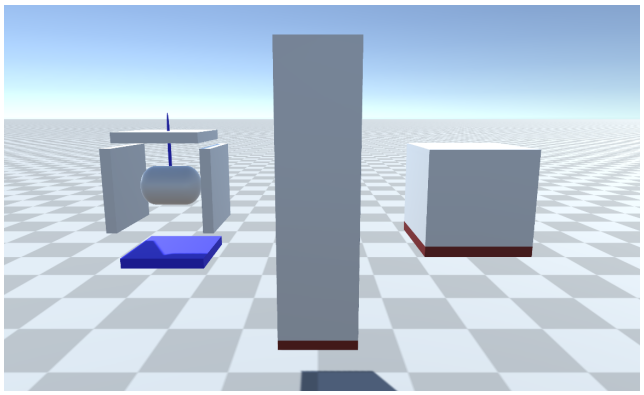

Figure 2: The different modules, from left to right: SEA (serial elastic actuator), beam, connector.

Our contributions are threefold, 1) we evolve modular robots with joints with variable compliance, and find that the robots evolve to use elastic actuators. 2) We explore the use of structural components, and observe that they can replace a large portion of the actuators, without reducing the robots' walking speed. 3) We do an initial investigation into connections between elasticity and size in modular robots, which indicates that larger robots may require stiffer springs.

Methods

In this work we perform two experiments where we 1) evolve modular robots with a standard evolutionary algorithm, and look at the emergence of elasticity, and 2) look at the usefulness of springs in modular robots by illuminating the search space of modular robots using MAP-Elites. For both experiments we use a modular robot simulator developed in Unity, using Unity ML-agents (Juliani et al., 2020). The following sections will describe 1) details of the modular robot simulator, 2) the encoding of the robots' morphology and controller. The same encoding is used in both experiments. 3) measures we use to evaluate the robots, 4) the standard evolutionary algorithm used in experiment 1, and 5) the MAP-Elites approach used in experiment 2. [1]

Simulator

We use a version of the modular robot simulator used in (Kvalsund et al., 2022). We have extended this simulator to allow for serial elastic actuators with evolvable elasticity. The modular robot simulator is developed in unity and communicates with Python through the ML-agents package. It uses Nvidia's PhysX physics engine. During simulation a robot is spawned in the middle of a rugged floor. The distance it moves and the torque that is applied to its joints is

[1]Code is available at `https://github.com/EmmaStensby/modular-robots-sea`

Module	Length	Plates	Actuated	Weight
SEA (S)	1	3	Yes	1
Beam (B)	3	1	No	1
Connector (C)	1	5	No	0.33

Table 1: Information about the robot modules. Plates refers to the number of connection plates where a child module can be connected.

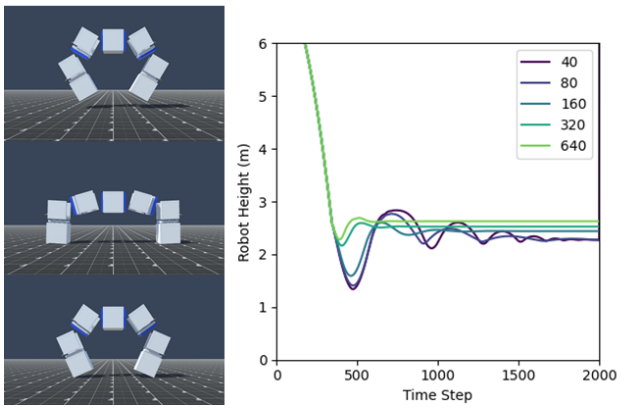

Figure 3: The bounceback of a test robot dropped from 7 meters for different values of the spring constant. The compression and bounceback is higher for lower spring values.

recorded. Some examples of modular robots in the simulator are shown in Fig. 1.

The simulation of a robot lasts 10 seconds. The robot makes a control decision for each update in the physics engine. We use an accuracy of 0.002 for the physics simulation, which means that one physics update happens every 0.002 seconds, for a total 5000 physics updates per simulation. Each physics update has an accuracy of 200 solver iterations. This high accuracy was chosen to be able to simulate the springiness in the serial elastic actuators. Fig. 3 demonstrates the effect of the elastic actuators in the simulation.

The simulator takes in a tree representing a robot, and spawns the robot starting with the modules closest to the root. The root module is spawned at a height of 3 units above the floor, where 1 unit is the length of a joint module. If a module overlaps with a previously spawned module it, and all its children, are removed. Any modules spawned below the floor or above a height of 9 units are also removed along with their children. There are 3 module types, the serial elastic actuator (SEA), a connector block, and a beam. The details of the modules are summarized in Table 1, and the modules are displayed in Fig. 2. The floor is slightly rugged, with a maximum height difference of 0.16 units between the lowest and highest points of the floor.

Amplitude	$0.0, 11.25, 22.5, 33.75, 45, 56.25, 67.5, 78.75, 90.0$
Frequency	$0.0625, 0.125, 0.25, 0.5, 1.0$
Phase	$0.125\pi, 0.25\pi, 0.5\pi, \pi, 2\pi$

Table 2: The possible values for the discretized sine controller parameters. The amplitude is in degrees of rotation for the motor.

Encoding

We use a direct encoding. The phenotype of a robot is a tree, where each node in the tree directly represents one module. The node contains information about the module type, and the module's rotation. If the module is actuated the node also contains information about the module's spring constant and controller. The edges of the tree define how the modules will be connected. Each node can have as many children as its module has connection plates.

Controller The controller for a module is a sine wave defined as:

$$\theta(t) = A\sin(2\pi ft + \phi) \tag{1}$$

Where θ is the desired joint angle, A is the amplitude, f the frequency, and ϕ the phase offset. One controller is stored in each node of that represents an actuated module. The possible values for the sine wave parameters are discretized to make it easier for the robot modules to synchronize. The possible values for each parameter are summarised in Table 2.

Spring stiffness The parameter for the spring constant is stored in each node representing an actuated module, and is a float between 0 and 1. However, because the effect of changing it is higher when the value is low, we use a natural logarithmic scale. The parameter is translated to the spring constant as:

$$S = \frac{(1000(e^{es-e})) - 65}{1000 - 65}1000 \tag{2}$$

Where s is the spring constant parameter. This gives a natural logarithmic scale from approximately 1 to 1000 for the spring constant.

Mutation A newly initialized tree, representing a robot, starts out with one module. It can be mutated to add or remove nodes, or to modify a node's spring constant or controller. The module type can not be modified. The mutation function iterates over all nodes in the tree.

The probability of adding a node to an empty connection is the morphology mutation rate (see Table 3) divided by the number of modules in the tree. Newly added nodes also have a chance to be mutated. The probability of removing a node is 0.5 times the probability of adding a node. When a node is

removed all its children are also removed. If the maximum number of modules is reached new nodes can not be added. The root node can never be removed.

The probability of modifying the spring constant or the controller parameters is the controller mutation rate divided by the number of joints in the robot. The probability of mutation is applied separately for each parameter. The spring constant is mutated by adding Gaussian noise drawn from $N(0, 0.3)$. The discretized controller parameters are replaced by a new value drawn uniformly from the parameter's possible values.

When a new tree is initialized, it is mutated 40 times to ensure some diversity in the starting population.

Robot evaluation measures

Distance moved The distance moved is used as the fitness measure in all our experiments. It is measured as the difference in start and end position of the root module of the robot. Due to issues with falling robots, any robot whose root module is rotated more than 80 degrees around either the X or Z axes is assigned a distance of 0.

Cost of transport The cost of transport is not used as a fitness measure for either of the evolutionary algorithms, but is used to evaluate the efficiency of the found solutions after the experiments are completed. We define the cost of transport in one simulation step as:

$$COT = \frac{T}{mgv} \tag{3}$$

Where T is the torque of the motors, which is read from the unity simulation. m is the mass of the robot, g is the gravity constant, which is set to -19.632, and v is the speed of the robot. Gravity is set to -19.632 to avoid excessive jumping behaviour. The speed of the robot is measured as the average speed of all the modules in the robot. The cost of transport is summed for each time step to form the final cost of transport.

Standard EA

The standard evolutionary algorithm (Eiben and Smith, 2015) uses tournament selection as the parent selection, and generational replacement as the survivor selection. The parameters used can be seen in Table 3. The mutation rates and tournament size were chosen based on the parameters used in a different work using the same simulator, to avoid spending a lot of computation on parameter search (Kvalsund et al., 2022).

MAP-Elites

We use the standard version of MAP-Elites as first described by (Mouret and Clune, 2015). We choose to use the standard version, as opposed to existing and potentially more efficient extensions, as we want to focus on a simple approach with an intuitive map structure to illuminate the design properties of the robotic system.

Standard EA	
Tournament size	4
Population size	64
MAP-Elites	
Map resolution	20
Map dimensions	2
Individuals evaluated per iteration	64
Shared	
Morphology mutation rate	0.32
Controller mutation rate	0.64
Maximum modules	20
Maximum depth	10

Table 3: Parameters for the standard evolutionary algorithm and MAP-Elites.

MAP-Elites generates a repertoire of elite individuals, each filling a niche of a discretised behaviour space. Every iteration of the algorithm individuals are selected randomly from the repertoire, mutated, evaluated, and then reinserted into the repertoire if they perform better than the existing individual in their niche. Only one individual is stored for each niche at any time. The discretisation of the behaviour space is defined through behaviour dimensions, which serve as the axes of the grid shaped repertoire. We use the following behaviour dimensions in our map:

1. The number of modules in the robot
2. The average spring constant in the SEA joints (0 if no joints)

The dimensions are divided into 20 bins, giving a two-dimensional map with 400 cells. The number of modules ranges from 1 to 20 as the maximum number of modules is 20. The spring constant ranges from 1 to 1000. Because the scale of the spring constant is non-linear, the bin sizes are also non-linear. The bins are defined by:

$$B_i = \frac{(1000(e^{e^{\frac{b_i}{b_m}}-e})) - 65}{1000 - 65} 1000 \tag{4}$$

Where b_i is the bin index and b_m is the total number of bins. Bin i goes from B_{i-1} to B_i, for bins 1 to 20.

Results

Experiment 1: Evolutionary use of springs and structural components

In the first experiment we wanted to explore the use of springs and structural components in a standard evolutionary algorithm. We did 4 sets of 20 runs of the EA described previously. The length of the runs was 128 generations, which

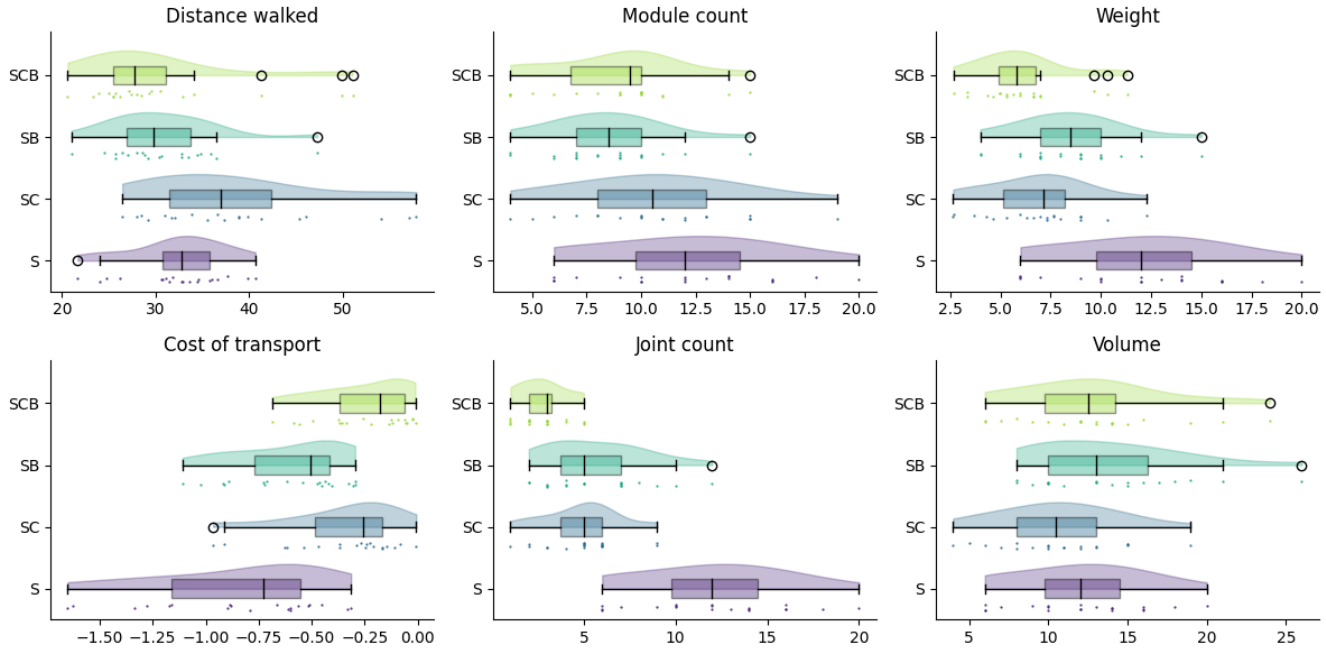

Figure 4: Behavioral and morphological measures of the results of the single-objective EA when evolved with *Distance walked* (top left) as the objective. The letters refer to the four different evolution scenarios with different available building blocks: S refers to the serial elastic actuator, C to the connector block, and B to the beam.

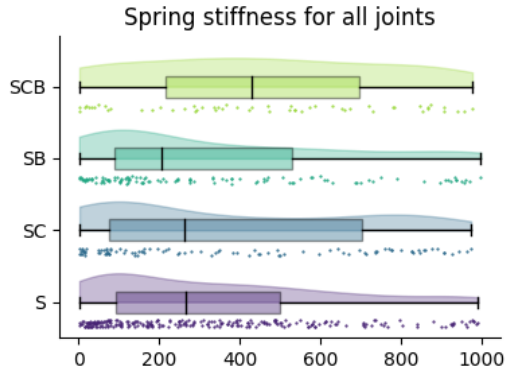

Figure 5: Spring constant values from all actuators in the best solution of each of the 20 runs, when using the distance-based single objective EA. The letters refer to the constituents of the four different sets of modules available to the EA: serial elastic actuator (S), connector block (C), and beam (B). The spring constant scale in this graph is linear.

corresponds to 8192 evaluated individuals. The spring constants for the joints were evolved in all four sets, but in each set the algorithm had access to different modules. We label the four sets S, SC, SB and SCB, based on which modules were used. S refers to the serial elastic actuator, C to the connector block, and B to the beam. Statistics from this experiment are shown in Fig. 5 and 4, and examples of typical morphologies evolved for each set are shown in Fig. 1.

From Fig. 4 we can see that robots with about the same fitness evolved for all four combinations of modules. The combinations with structural components had a lower cost of transport than the one with only joints, likely because fewer motors, and thus less energy, was used. The cost of transport was lowest for the two combinations that included the connector block. The robots using the connector block also had lower weight, so it might seem like this module enables some morphologies that have both lower weight and lower cost of transport.

The combinations with structural modules had significantly fewer joints, as structural modules were used instead, and the combinations with beams seem to have created robots with slightly fewer modules, likely because the beam has fewer connection points than the other modules.

From Fig. 5 we can see that S, SC and SB use modules with a low spring constant a bit more frequently than modules with a high spring constant. For SCB the modules are spread more evenly across the spring constant values, which

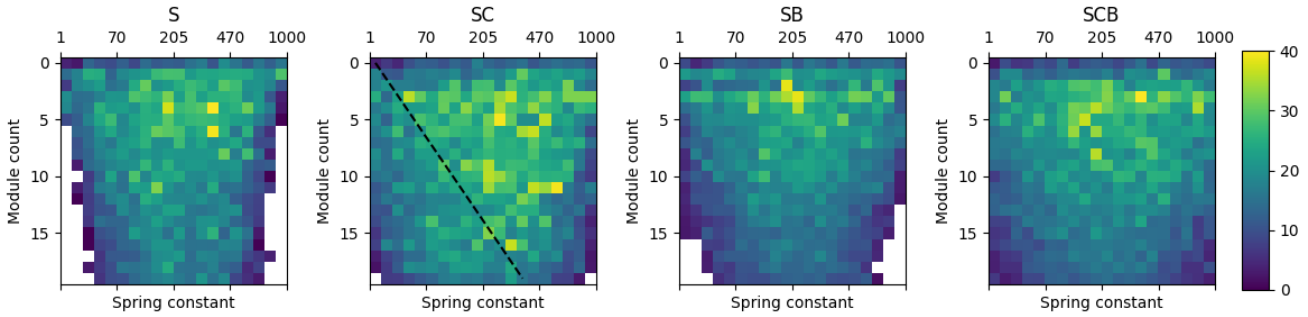

Figure 6: Repertoires from MAP-Elites when evolved with *Distance walked* as the objective. The color is the highest fitness found for each cell over all 20 runs. White cells indicate no robots were found for that cell. The letters refer to the four different evolution scenarios with different available building blocks: S refers to the serial elastic actuator, C to the connector block, and B to the beam.

may be connected to the low number of joints it uses.

In Fig. 1 we can qualitatively evaluate a few morphologies, and see that the robots used all module types they had available to construct the morphologies. Although the sample size is small, it may seem like the morphologies using only joints have slightly longer worm-like bodies, while the ones using the connector block have slightly more compact bodies. [2]

Experiment 2: Relations between spring stiffness and robot size

In the second experiment, we wanted to explore relations between robot size and spring stiffness using the quality-diversity algorithm MAP-Elites. We did 4 sets of 20 runs. The four sets use the same combinations of modules as the four sets from experiment 1. The length of the runs was 256 generations, which corresponds to 16384 evaluated individuals.

The evolved repertoires are displayed in Fig. 6. We can see that all combinations of modules evolved high fitness robots, but the set using the joint and connector found high fitness for a larger variety of descriptor combinations. We also see that the high fitness solutions lie mostly in the top of the repertoire, where the small robots are, for S, SB and SCB. For SC the high fitness solutions lie mostly in the top right half of the repertoire. For SC this is indicated with a black dashed line.

The white areas indicate no robots were found for those descriptor combinations. The white areas are the most prominent for large robots, and for the combinations that tended to use many joints. We therefore believe the white areas stems from it being difficult to reach solutions with the same elasticity in many motors. This could be due to high mutation rate, as high mutation rate would pull the average of the spring constants towards the center.

Discussion

From Fig. 4 we saw that evolution finds solutions with approximately the same fitness regardless of the modules available. This is despite the robots evolved with access to structural modules, that is, the connector block and the beam, having significantly fewer joints to move with. The robots with structural modules have accomplished the same task, but with fewer actuators. This shows that including structural components can be an advantage. The evolution scenarios that included the connector block had lower robot weight despite having lower cost of transport. This likely means that the evolution has found a way to use less motor torque by using the connector block. Either indirectly through the lower weight, or because of some other property, like the numerous connection points.

Figure 5 shows that joint modules with elasticity were used a bit more frequently than stiffer joints for S, SC and SB in the standard EA runs. This may indicate usefulness of the elasticity, although it may also be an artefact of the non-linearity of the spring constant scale during evolution. For SCB the distribution of the elasticity in the motors was more uniform than for the other three combinations, meaning that it comparatively used less elastic joints.

SCB used very few joints, but still had equal volume to the other combinations. This may indicate that stiffer joints could be more useful when a few actuators are supporting a large robot. Although it is difficult to say for certain without more data, this would be in line with the observations on animals from (Farley et al., 1993). The MAP-Elites repertoire for SC in Fig. 6 could also support larger robots requiring stiffer joints. No high fitness solutions have been found below the dashed line, while several have been found above. The cells below the line hold the robots that have very loose springs relative to their size. We also saw from the reper-

[2] Videos of some robots are available at `https://youtu.be/9uRyqwFZhv0` and `https://youtu.be/yH7CaY8Qqq4`

toires that SC had found high fitness for more diverse morphologies than the other three setups, indicating that SC may be the best setup of components out of the four we evaluated.

It is difficult to evaluate from the resulting robots whether the springs were used in the intended way, to store and release kinetic energy. Our lack of clear observations on morphological differences is in line with previous research (Miras and Eiben, 2019), and may be related to the evolutionary algorithm being influenced by many factors. The simultaneous evolution of morphology and controller is quite complex, and many other factors may have more effect on the results than the elastic property we wanted to study.

Conclusion

In this paper we investigated how compliant and structural modules were used, by an evolutionary algorithm, when evolving modular robots. We also investigated connections between robot morphology and elasticity. We found that the modular robots evolved to use elastic actuators and structural modules, without causing a reduction in walking speed, and did an initial investigation into connections between elasticity and size in modular robots, which indicated that larger robots require stiffer springs. In the future we would like to further explore connections between morphological features and elasticity, and test the evolved modular robots on a physical platform in the real world.

Acknowledgements

This work was partially supported by the Research Council of Norway through its Centres of Excellence scheme, project number 262762. The simulations were performed on resources provided by UNINETT Sigma2—the National Infrastructure for High Performance Computing and Data Storage in Norway.

References

Alexander, R. M. (1984). Elastic energy stores in running vertebrates. *American Zoologist*, 24(1):85–94.

Auerbach, J., Aydin, D., Maesani, A., Kornatowski, P., Cieslewski, T., Heitz, G., Fernando, P., Loshchilov, I., Daler, L., and Floreano, D. (2014). RoboGen: Robot Generation through Artificial Evolution. *Artificial Life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems*, pages 136–137. ISBN: 9780262326216.

Auerbach, J. E. and Bongard, J. C. (2014). Environmental influence on the evolution of morphological complexity in machines. *PLoS computational biology*, 10(1):e1003399.

Biewener, A. (2003). *Animal locomotion*. Oxford University Press.

Blackiston, D., Lederer, E., Kriegman, S., Garnier, S., Bongard, J., and Levin, M. (2021). A cellular platform for the development of synthetic living machines. *Science Robotics*, 6(52):eabf1571.

Chatzilygeroudis, K., Cully, A., Vassiliades, V., and Mouret, J.-B. (2021). Quality-diversity optimization: a novel branch of stochastic optimization. In *Black Box Optimization, Machine Learning, and No-Free Lunch Theorems*, pages 109–135. Springer.

Cheney, N., Clune, J., and Lipson, H. (2014). Evolved electrophysiological soft robots. In *Artificial life conference proceedings*, pages 222–229. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info ....

Cheney, N., Sunspiral, V., Bongard, J., and Lipson, H. (2016). On the difficulty of co-optimizing morphology and control in evolved virtual creatures. In *Artificial Life Conference Proceedings 13*, pages 226–233. MIT Press.

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

Doncieux, S., Mouret, J.-B., Bredeche, N., and Padois, V. (2011). Evolutionary robotics: Exploring new horizons. In *New Horizons in Evolutionary Robotics*, pages 3–25. Springer.

Eiben, A. E. and Smith, J. E. (2015). *Introduction to evolutionary computing*. Springer.

Faíña, A., Bellas, F., López-Peña, F., and Duro, R. J. (2013). Edhmor: Evolutionary designer of heterogeneous modular robots. *Engineering Applications of Artificial Intelligence*, 26(10):2408–2423.

Farley, C. T., Glasheen, J., and Mcmahon, T. A. (1993). Running Springs: Speed and Animal Size. *Journal of Experimental Biology*, 185(1):71–86.

Juliani, A., Berges, V.-P., Teng, E., Cohen, A., Harper, J., Elion, C., Goy, C., Gao, Y., Henry, H., Mattar, M., and Lange, D. (2020). Unity: A general platform for intelligent agents. *arXiv preprint arXiv:1809.02627*.

Kalouche, S., Rollinson, D., and Choset, H. (2015). Modularity for maximum mobility and manipulation: Control of a reconfigurable legged robot with series-elastic actuators. In *2015 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*, pages 1–8.

Kvalsund, M.-K., Glette, K., and Veenstra, F. (2022). Centralized and decentralized control in modular robots and their effect on morphology. In *ALIFE 2022: The 2022 Conference on Artificial Life*. MIT Press.

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

Liu, C., Liu, J., Moreno, R., Veenstra, F., and Faina, A. (2017). The impact of module morphologies on modular robots. In *2017 18th International Conference on Advanced Robotics (ICAR)*, pages 237–243.

Miras, K. and Eiben, A. E. (2019). Effects of environmental conditions on evolved robot morphologies and behavior. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '19, page 125–132, New York, NY, USA. Association for Computing Machinery.

Miras, K. and Ferrante, E. (2020). Environmental influences on evolvable robots. *PloS one*, 15(5):e0233848.

Miras, K., Haasdijk, E., and Glette, K. (2018a). Search space analysis of evolvable robot morphologies. In *Applications of Evolutionary Computation: 21st International Conference, EvoApplications 2018, Parma, Italy, April 4-6, 2018, Proceedings 21*, pages 703–718. Springer.

Miras, K., Haasdijk, E., Glette, K., and Eiben, A. (2018b). Effects of selection preferences on evolved robot morphologies and behaviors. In *Artificial Life Conference Proceedings*, pages 224–231. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info ....

Moreno, R. and Faina, A. (2020). Using evolution to design modular robots: An empirical approach to select module designs. In Castillo, P. A., Jiménez Laredo, J. L., and Fernández de Vega, F., editors, *Applications of Evolutionary Computation*, pages 276–290, Cham. Springer International Publishing.

Mouret, J.-B. and Clune, J. (2015). Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*.

Nordmoen, J., Veenstra, F., Ellefsen, K. O., and Glette, K. (2021). MAP-Elites enables powerful stepping stones and diversity for modular robotics. *Frontiers in Robotics and AI*, 8:56.

Pratt, G. A. and Williamson, M. M. (1995). Series elastic actuators. In *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, volume 1, pages 399–406. IEEE.

Pratt, J. E. and Krupp, B. T. (2004). Series Elastic Actuators for legged robots. In Gerhart, G. R., Shoemaker, C. M., and Gage, D. W., editors, *Unmanned Ground Vehicle Technology VI*, volume 5422, pages 135 – 144. International Society for Optics and Photonics, SPIE.

Roberts, S. F., Hirokawa, J., Rosenblum, H. G., Sakhtah, H., Gutierrez, A. A., Porter, M. E., and Long Jr, J. H. (2014). Testing biological hypotheses with embodied robots: adaptations, accidents, and by-products in the evolution of vertebrates. *Frontiers in Robotics and AI*, 1:12.

Sims, K. (1994). Evolving virtual creatures. *Siggraph '94*, SIGGRAPH '(July):15–22. ISBN: 0897916670.

Stoy, K., Brandt, D., and Christensen, D. J. (2010). *Self-Reconfigurable Robots: An Introduction (Intelligent Robotics and Autonomous Agents series*.

Veenstra, F. and Glette, K. (2020). How different encodings affect performance and diversification when evolving the morphology and control of 2d virtual creatures. In *ALIFE 2020: The 2020 Conference on Artificial Life*, pages 592–601. MIT Press.

Yim, M., Shen, W.-M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., and Chirikjian, G. S. (2007). Modular self-reconfigurable robot systems. *IEEE Robotics & Automation Magazine*, 14(1):43–52.

Yu, H., Huang, S., Chen, G., Pan, Y., and Guo, Z. (2015). Human–robot interaction control of rehabilitation robots with series elastic actuators. *IEEE Transactions on Robotics*, 31(5):1089–1100.

Zhao, A., Xu, J., Konaković Luković, M., Hughes, J., Speilberg, A., Rus, D., and Matusik, W. (2020). Robogrammar: Graph grammar for terrain-optimized robot design. *Transactions on Graphics*, 39(6):1–16.

Zhou, X. and Bi, S. (2012). A survey of bio-inspired compliant legged robot designs. *Bioinspiration & Biomimetics*, 7(4):041001.