Severe damage recovery in evolving soft robots through differentiable programming

Kazuya Horibe1 · Kathryn Walker2 · Rasmus Berg Palm2 · Shyam Sudhakaran2 · Sebastian Risi2

Received: 1 November 2021 / Revised: 31 January 2022 / Accepted: 28 March 2022 / Published online: 12 June 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract
Biological systems are very robust to morphological damage, but artificial systems (robots) are currently not. In this paper we present a system based on neural cellular automata, in which locomoting robots are evolved and then given the ability to regenerate their morphology from damage through gradient-based training. Our approach thus combines the benefits of evolution to discover a wide range of different robot morphologies, with the efficiency of supervised training for robustness through differentiable update rules. The resulting neural cellular automata are able to grow virtual robots capable of regaining more than 80% of their functionality, even after severe types of morphological damage.

Keywords Regeneration · Soft robots · Neural cellular automata · Damage recovering

Kathryn Walker, Rasmus Berg Palm and Shyam Sudhakaran have contributed equally to this work.

✉ Kazuya Horibe
horibe@irl.sys.es.osaka-u.ac.jp
Kathryn Walker
kwal@itu.dk
Rasmus Berg Palm
rasmb@itu.dk
Sebastian Risi
sebr@itu.dk

1 Department of Systems Innovation, Osaka University, 1-3 Machikaneyama, Toyonaka 560-8531, Osaka, Japan
2 Department of Digital Design, IT University of Copenhagen, Rued Langgaards Vej 7, Copenhagen DK 2300, Denmark
1 Introduction

Within the natural world, evolution has created a diverse range of robust, adaptive organisms that are able to survive damage. While this adaptation is sometimes due to a change in behaviour (or control system), in many cases these organisms rely on morphological regeneration. In these instances, the organisms repair and/or reconfigure their morphology in response to damage or changes in components [1].

The amount of regeneration for which an organism is capable varies from species to species. For many years, gardeners have deliberately damaged (pruned) their plants to encourage plant re-growth in a particular direction or increase fruit yield [2, 3]. Plant propagation through cutting (where a completely new plant is grown from part of an existing plant’s stem or root) is also common practice [4]. Whilst pruning can be considered a chore by many [2], it does help to highlight the power of nature’s morphological regeneration.

Morphological regeneration does not only occur in plants but also in animals. For instance, salamanders are capable of regenerating an amputated leg [5]. The simple organisms Hydra and Planaria are capable of complete morphological repair, regardless of which body part is removed [6, 7]. Even within humans, the liver organ is able to regenerate and replace lost liver tissue via growth from the remaining cells [8].

In contrast to biological organisms, artificial robotic systems are fragile and even a small amount of damage can severely impact their performance. Furthermore, despite its commonplace within nature, the majority of examples of robots adapting to damage focus on a change in control system, not morphology. This potentially limits their robustness compared with the wide range of damage natural systems are able to recover from.

While the mechanisms behind natural morphological regeneration are still not fully understood, artificial methods such as neural cellular automata have proven successful tools for their modelling. For instance, the work presented in this paper is an extension of our previous conference paper [9], where we evolve neural cellular automata to grow locomoting soft robots capable of some morphological regeneration. Neural cellular automata are based on the simpler cellular automata. These consist of a regular grid of cells where each cell can be in any one of a finite set of states, cell states are then updated based on information from their neighbour’s states and simple rule sets. In neural cellular automata, these simple rules are replaced by artificial neural networks.

In this paper, we extend our initial investigation and use a more efficient way of training neural cellular automata for robustness, which is based on gradient-based optimization. An overview of the approach is shown in Figure 1. As before [9], we first use an evolutionary algorithm to discover neural cellular automata that can grow the control and morphology of a diversity of soft voxel-based robots. The NCA for a particular robot is then efficiently trained through differentiable programming (i.e. gradient-based optimization) to grow a particular robot while being able to recover from various forms of damage.
In contrast to previous work on training for regeneration through gradient-based optimization [10, 11], here the target artefacts are evolved instead of being hand-designed. This approach thus combines the creativity of artificial evolution with the efficiency of supervised training, opening up new application in the field of morphological computation.

1.1 Related work

1.1.1 Evolving virtual creatures

In this paper we explore growing virtual soft robots, capable of regeneration, via the use of trained neural cellular automata. In the first instance we use evolutionary algorithms to grow simple soft robots, using distance travelled as our fitness function. However, the concept of using artificial evolution to design the control and morphology of virtual robots has existed for almost three decades. In his seminal work, Karl Sims evolved the body plans of simple block based robots, which interacted with their virtual environment [12, 13]. This work was then followed by many researchers also investigating using evolutionary algorithms to design the morphology and control of virtual robots, e.g., [14–18].

A common theme amongst researchers investigating evolving robot morphologies is the use of compositional pattern producing networks (CPPNs) to describe body shape and properties. Popularised by researchers such as Stanley [19], Bongard and Cheney [20–24], CPPNs are a special type of artificial neural network, applied over the whole input space, allowing patterns or shapes to be produced.

However, these previous examples only consider static morphologies, that is, they do not allow for shape change in any way, be that growth or damage recovery. They are not robust to changes in the environment or damage. Therefore, more recently, researchers have begun to investigate these areas; these are described in the section below.
1.1.2 Damage recovery/shape change in robots

As previously discussed, when trying to adapt to new environments/tasks or recover from damage, researchers have experimented with changing with control system of the robot, not the morphology, e.g. [25–30]. In contrast, the investigation into metamorphosis/morphological regeneration as damage recovery strategy is less common, despite its potential and prevalence in nature.

Recent examples of robots adapting their morphology, rather than control, to different environments include the work by Kriegman et al. [31] where individual parts of the robot change stiffness in response to external stresses. Walker et al. [32] explore removing individual parts of the robot to sculpt morphologies based on environmental interaction. Shah et al. published work on a physical soft robot capable of radial morphological shape change for locomotion in different environments [33]. Additionally, Shah et al. published a review paper of shape changing robots for environmental adaptation - refer to this for a fully comprehensive literature review [34].

These previous examples all refer to morphology change for environmental adaptation. However, there are also examples where morphology change has been used for damage recovery. These works include that by Kriegman et al. [35], where silicone based physical voxel robots were able to recover from voxel removal. Furthermore, Xenobots, synthetic creatures designed from biological tissue [36], have shown to be capable of morphological reattachment (i.e. healing after insult).

1.1.3 Neural cellular automata for growth and recovery

Cellular automata (CA) were first proposed by Neumann and Ulam in the 1940s and, as previously discussed, consist of a regular grid of cells where each cell can be in any one of a finite set of states [37]. Each cell determines its next state based on local information (i.e. the states of its neighboring cells) according to pre-defined rules. In a neural cellular automata, instead of having hand-designed rules, a neural network learns the update rules [38].

Neural cellular automata have been used to great effect for morphological growth and recovery in simulation. For example, in 2004, Miller showed that automatic recovery of simple damaged target patterns is possible with genetic programming [38]. More recently, Mordvintsev et al. trained a neural CA to grow complex two-dimensional images starting from a few initial cells through gradient-based optimization (i.e. differentiable programming) [11]. In addition, the authors have also successfully trained the system to reconstruct the pattern after damage (i.e. it was able to regrow the target pattern). The neural network in their work is a convolutional network, which lends itself to represent neural CAs [39].

Based on Mordvintsev’s work, Sudhakaran et al. [10] have trained neural cellular automata capable of growing particular target patterns in three dimensions. As with Mordvintsev’s work, these 3D shapes/structures are able to regrow after
significant damage and are also trained through gradient-based optimization. Both approaches [10, 11] relied on user-specific target patterns, while these target patterns are themselves evolved in our approach and then made robust against damage.

2 Method

2.1 Soft robot simulator

For our investigations (both 2D and 3D) we use the soft robot simulator “Voxelyze” [40]. In Voxelyze, a cell takes the form of a 3D cube called a voxel; adjacent voxels (or cells) are connected together by simulated beams. Voxelyze creates actuation within these soft robots by employing a sinusoidally varying global control signal (termed temperature by the Voxelyze software). Active cells expand and contract in phase with this sinusoidal temperature variation, whilst passive cells remain a constant size. The software VoxCad can then be used to visualise this.

Our aim is to evolve soft robots capable of successful locomotion, as well as regeneration, both in 2D and 3D. Thus the fitness of a robot (in terms of locomotion) is how far the robot travelled in a fixed time period. This time period is 0.25 seconds, or 10 actuation cycles in 2D and for 0.5 seconds, or 20 actuation cycles in 3D. The evaluation times try to strike a balance between reducing computational costs while still giving sufficient time to observe interesting locomotion behaviours. Fitness is determined as the distance the robot’s center of mass moves in 0.25 or 0.5 seconds. The distance (or fitness) of a robot is measured in units that correspond to the length of a voxel with volume one. It should also be noted that creatures with zero voxels after their growth are automatically assigned a fitness of 0.0.

In all our experiments, the neural cellular automata is able to utilise four types of cell/voxel. The first is an active cell, denoted by the colour red, and has an expansion and contraction cycle in phase with the global control signal. We also refer to this type of cell as muscle. The second type of cell can also be thought of as muscle. Denoted by the colour green, it also expands and contracts, but this time at counter phase to the red muscle cells. Dark blue cells are passive, they do not change size. We refer to these as bone. The final cell type is denoted by a light blue colour. These are also passive but have a lower stiffness than dark blue bone voxels.

Our choice of cell types is consistent with previous literature (Cheney et al. [20]), as is our choice of physical and environmental Voxelyze parameters.

2.2 Neural cellular automata

As previously discussed, Neural Cellular Automata (NCA) are Cellular Automata in which each cells state is represented by a vector of scalar values, and the update rules are parameterized by neural networks [11]. Our cell state vector consists of three parts: 1) The number of unique output types for each cell $k$, e.g. muscle, bone, etc., corresponds to the first $k$ channels, 2) the cell aliveness value and 3) a number of hidden states. The NCA grid is seeded with a single central cell at $t = 0$ with its state vector set
to all ones and all the other cell states to zero. The NCA update rule defines a recurrent additive function parameterized by a neural network,

$$h_i^t = a_i^t \cdot (h_i^{t-1} + u_i^t \cdot f_\theta(h_i^{t-1}_{j \in N(i)})),$$

where $h_i^t \in \mathbb{R}^D$ is the $D$ dimensional state of cell $i$ at time $t$, $f_\theta$ is the update rule, parameterized by a neural network with parameters $\theta$ and $N(i)$ is the set of indices of all the neighbors of cell $i$, including $i$. For a NCA defined on a 3D grid it corresponds to the cells in a $3 \times 3 \times 3$ neighborhood [10]. $a_i^t \in \{0, 1\}$ and $u_i^t \in \{0, 1\}$ are the alive and update masks respectively.

The update mask is a random binary variable sampled with some probability $p$, so that $p(u_i^t = 1) = p$. The random update mask can be seen as a form of dropout [41] which makes the NCA robust to noise and invariant to the exact order of the updates. In our experiments we use $p = 0.5$.

The aliveness mask ensures that dead cells have an all zero state. A cell is considered alive if any of the cells in its $3 \times 3 \times 3$ neighborhood, including itself, have an aliveness value of more than 0.1. The aliveness value is the $k$'th entry of the cell state vector, so that $a_i^t = \max(h_i^{t-1}[k]_{j \in N(i)}) \geq 0.1$. The aliveness mask is efficiently computed using a 3D max pooling operation.

### 3 Evolving soft robot morphologies

To confirm the promise of NCA for growing soft robots, we use a simple three-layer networks with tanh activation functions (Fig. 2a). We experiment with both feed forward (the hidden layer is a linear layer) and recurrent networks (the hidden layer is an LSTM unit [42]), which means that each cell has its own memory. Here, the dimension of the hidden layer is set to 64 unless otherwise noted. The recurrent setup is inspired by recent experimental reports that organisms store information about the original morphology in a distributed manner in the bioelectrical signaling networks [43, 44].

To evolve a NCA for initial soft robot growth, we use a simple genetic algorithm [45, 46] that has been shown to be effective in training deep neural networks [47, 48]. The implemented GA variant performs truncation selection with the top $T$ individuals becoming the parents of the next generation. The following procedure is repeated at each generation: First, parents are selected uniformly at random. They are mutated by adding Gaussian noise to the weight vector of the neural network (its genotype): $\theta' = \theta + \sigma \epsilon$, where $\epsilon$ is drawn from normal distribution $N(0, I)$ and $\sigma$ is set to 0.03. Following a technique called elitism, top $M^{th}$ individuals are passed on to the next generation without mutation.

#### 3.1 2D soft robots

We first investigate the use of NCA in 2D. Here, robots have a maximum size of $7 \times 7$ voxels. Since the NCA used a Moore neighborhood, the input dimension of the neural network was $9 \times 2 = 18$, which includes neighboring cell types and alpha values.
The output layer of the NCA has a size of 6 neurons; 5 neurons for the different states of the cell (empty=0, light blue=1, dark blue=2, red=3, green=4) plus one alpha channel. The first single soft & passive cell (light blue) is placed at position (3, 3) and 10 steps of development are performed. As result, 11 morphologies are obtained, see (Fig. 3a). Afterwards, the final grown robot is tested in the physical simulator (Voxelyze) and allowed to attempt locomotion for 0.25 seconds, i.e., 10 actuation cycles. The fitness of each robot is taken to be distance travelled by the robot from its starting point. These 2D experiments use a population size of 300, running for 500 generations. One evolutionary run on 8 CPUs took around 12 hours.

Results are shown in Fig. 3, which were obtained from ten independent evolutionary runs, using both recurrent and feed forward networks. The training mean together with bootstrapped 95% confidence intervals is shown in Fig. 3b.

Evolution produced a variety of soft robots (Fig. 3c). A “Hook” type is distinguished by its hook-like form and locomotion, which shakes the two sides of the hook and the proceeds to hook the remaining one side to the floor. The “S” shaped robot is distinguished by its sharp and peristaltic motion with amplitude in the same direction as the direction of travel. The “Biped” has two legs and its locomotion
resembles that of a frog, with the two legs pushing the robot forward. The “L” type displays a sharp and winged movement. Finally, the “Zig-zag” shows a spring-like movement by stretching and retracting the zigzag structure. Enabling the cells to keep a memory of recent developmental states through a recurrent network improved performance, although only slightly (Fig. 3b). Investigating what information the evolved LSTM-based network is keeping track of during development is an interesting future research direction.

3.2 3D soft robots

In this section we now extend our methodology to grow 3D robots. For these 3D robots the maximum size of the morphologies is $9 \times 9 \times 9$. Since again the NCA uses a Moore neighborhood, the input dimension of the neural network is $3 \times 9 \times 2 = 54$. The hidden layer is set to 64. The output layer is set to $5 + 1 = 6$ dimensions with the number of states of the cell and the value of its own next step alpha value. The first single soft & passive cell is placed at position $(4, 4, 4)$ and 10 steps of growth are performed (Fig. 4a). The final soft robot grown after 10 steps is tested in Voxelyze and, as with the 2D robots, the distance of the robot’s center of gravity from its starting point was used as part of the fitness function. Additionally we include a voxel cost in the fitness calculation: \( \text{Fitness} = (\text{Distance}) - (\text{VoxelsCost}) \). We added a “voxel” cost because preliminary results indicated that without this additional metric all the soft robots simply acquired a box-like morphology. Including the voxel cost metric increased diversity in the population. Note that voxel cost is the number of voxels that are neither empty nor dead.
For our 3D experiments the evaluation time is increased to 0.5s for 20 actuation cycles to adjust for the increased complexity of the robots. Each generation has a population size of 100 and the next generation is selected from the top 20%. The number of generations is set to 300. Note that both the generation number and population size are reduced from those values used in the 2D experiments as simulated the larger 3D robots has a higher computational cost. One evolutionary run on 1 CPU took around 80–90 hours.

Results are based on 24 independent evolutionary runs for both the recurrent and feed forward treatment (Fig. 4b). Interestingly, the feed forward setup for the 3D robots has a higher fitness than the recurrent one, in contrast to the 2D soft robot results (Fig. 3b). We hypothesize that with the increased numbers
of neighbors in 3D and more complex patterns, it might be harder to evolve an LSTM-based network that can use its memory component effectively. Because the dynamics of LSTM-based networks are inherently difficult to analyse, more experiments are needed to investigate this discrepancy further.

Similarly to CPPN-encoded soft robots [20], 3D robots grown by an evolved NCA (Figure 4) can be classified into two groups: the first group is the two-dimensional group of organisms (Fig. 4c), where planar morphology was acquired by evolution. Exemplary classes of locomotion in this group include the jumper, which is often composed of a single type of muscle voxel. Once a soft robot sinks down, it use this recoil to bounce up into the air and move forward. The morphology determines the angle of bounce and fall. The Roller is similar to a square; it moves in one direction by rotating and jumping around the corners of the square. The Push-Pull is a widely seen locomotion style. A soft robot pushes itself forward with its hind legs. During this push, it pulls itself forward, usually by hooking its front legs on the ground. The Slider has a front foot and a hind foot, and by opening and closing the two feet, it slides forward across the floor. The two legs are usually made of a single material. The Jitter moves by bouncing up and down from its hind legs to back. It has an elongated form and is often composed of a single type of muscle voxel. The second group is the three-dimensional group of organisms, as shown in Fig. 4d. The L-Walker resembles an L-shaped form; it moves by opening and closing the front and rear legs connected to its pivot point at the bend of the L. The Crawler has multiple short legs and its legs move forward in concert.

4 Training for regeneration

We now investigate the ability of the soft robots to regenerate their body parts to recover from morphological damage. We chose three morphologies from the previous experiments, which are able to locomote well and are as diverse as possible: the Biped (feed forward), Tripod (feed forward), and Multiped (recurrent). The morphologies of each of these three robots are shown in Fig. 5a and the locomotion patterns in Fig. 4d. We use two different approaches to train regeneration NCAs, the first using an evolutionary approach as previously reported in [9]. The second approach is based on gradient-based optimization, i.e. differentiable programming.

4.1 Evolutionary training for regeneration

In these experiments, which have been first reported in our conference paper [9] we damage the morphologies such that one side of the robot was completely removed (Fig. 5a). In the left side of these damaged morphologies, the cell states were set to empty and the maturation alpha values were set to zero. For the recurrent network, the memory of LSTM units in each cell were also reset to zero.
4.1.1 Network architecture and training method

We initially attempted regeneration using the original NCAs of these three robots but regeneration failed and locomotion was not recovered. Therefore, we evolved another NCA, where the sole purpose was to regrow a damaged morphology. In other words, one NCA grows the initial morphology and the other NCA is activated once the robot is damaged. Fitness for this second NCA is determined by the voxel similarity between the original morphology and the recovered morphology (values in the range of \([0; 729]\)). The maximum fitness of \(9 \times 9 \times 9 = 729\) indicates that the regrown morphology is identical to the original morphology. We evolved these soft robots, which were allowed to grow for 10 steps, for 1,000 generations with a population size of 1,000. The next generation was selected from the top 20%.

4.1.2 Morphology similarity and locomotion recovery

For all three morphologies we trained both feed forward and recurrent NCAs. The best performing network types for damage recovery were consistent with the original network type for locomotion in all morphologies (biped = feed forward, tripod = feed forward, multiped = recurrent). Training network for Biped (feed forward), Tripod (feed forward) and Multiped (recurrent) takes 45, 23, and 100 hours using single CPU core (2.7 GHz Intel Xeon E5). The results with the highest performing network type are summarised in Table 1 and damaged morphologies for each of the robots are shown in Fig. 5a. The results indicate that the Multiped was the hardest to reproduce, followed by the Biped and then the Tripod. The Tripod had a higher similarity than the other morphologies and the NCA almost completely reproduced the original morphology with the exception of one cell. We hypothesise that regeneration for the Tripod is easier because it only requires the regrowth of one leg, a simple rod-like shape with only a few cells.

For comparison, we then measured the locomotion of the original, damaged, and regrown morphology with an evaluation time of 0.5s for 10 cycles in Voxelyze. The ratio of regrowth and travel distance to the original morphology are shown in Table 1 and its locomotion in Fig. 9. The damaged Biped maintained 67% of its

| Morphology (Network) | Similarity | Locomotion |
|----------------------|------------|------------|
|                      | Original | Damaged | Regrown |
| Biped (feed forward) | 98% (718/729) | 40.4 | 27.2(67%) | 35.1(86%) |
| Tripod (feed forward) | 99% (728/729) | 44.5 | 1.63(3.6%) | 20.3(45%) |
| Multiped (recurrent) | 91% (667/729) | 42.7 | 5.36(12%) | 9.6(22%) |
original locomotion ability; it replicated a similar locomotion pattern to the one observed in the L-Walker. As the Tripod lost one of its three legs, it was incapable of successful locomotion. Furthermore, the Multiped lost all locomotion – the robot simply collapsed at the starting position.

These results suggest that the location of the damage is important in determining how much the robot loses in terms of locomotion performance. For instance, in the case of the Biped, the left hand side and right hand side are symmetrical. This means that when the left hand side was removed, the right hand side was able to

Fig. 5 Soft robots evolved for regeneration. a Original, damaged, and regrown morphology. b–d Soft robot development after damage shown at different timesteps. e Training performance for recurrent/feed forward setup
locomote in the same, almost unaffected way. Therefore, despite having the lowest similarity value between the initial and regrown morphologies, there is little loss in performance. In contrast the Tripod regained less than half the locomotion of the original morphology, despite regaining its original morphology almost completely. It would appear that the one voxel it is unable to regenerate is necessary to prevent the robot from spinning, allowing it to move forward.

Using this method of training two neural networks via an evolutionary approach shows potential for soft robotic regeneration using neural cellular automata. However, only one type of damage scenario is investigated, and even in this case our method is not successful at recovering the functionality of the robot. Therefore, in the next section, we investigate a new method of training one neural network for growth and recovery through gradient-based optimization.

4.2 Regeneration through differentiable programming

In the gradient-based training, one NCA is trained to grow a particular target structure (one of the three evolved robot morphologies) while being resilient to damage.

4.2.1 Network architecture and training method

Here we use a similar three-dimensional convolutional network than the one introduced by Sudhakaran et al [10]. This network (Fig. 2b) is an extension of the 3D network proposed by Mordvintsev et al [11], which has been shown to exhibit robust regeneration 2D patterns against various types of damage. The input of the NCA first passes through a perception net implemented with a 3D convolutional layer with kernel size=3, stride = 1 and output channels = cell state the NCA * 3. Next, its output passes through three linear 3D convolutional layers with kernel size=1, stride = 1 of the NCA update network. We apply the update to stochastic update according to the “dropout” to avoid over fitting [41]. And then, the output is followed by an “alive mask”. The “alive mask” is implemented by multiplying the MaxPool layer of the update by a boolean filter where 1 if the value of the living channel < 0.1, and 0 otherwise. We performed a standard normal initialization with standard normal = 0.1 and mean = 0.0 for all networks.

The additive update rule is inspired by residual networks [49], and helps with stabilizing the gradient during training. The update rule can be efficiently implemented using a single 3D convolution with a filter size of $3 \times 3 \times 3$ followed by a non-linearity and any number of 3D convolutions with $1 \times 1 \times 1$ filter sizes, followed by non-linearities. A final linear $1 \times 1 \times 1$ convolution maps back to the
NCA hidden state size. In our experiments we follow [10] and use two $1 \times 1 \times 1$ convolutions with 64 channels each, and ReLU non-linearities [50].

During training the NCA is run for a random number of steps, $T$ between 48 and 64, after which the loss is calculated. We use the loss from [10], which contains two parts,

$$\mathcal{L} = \sum_i \frac{1}{2} \text{CE}(y_i, t_i) + \text{IOU}(y_i, t_i)$$

where $y_i = \text{softmax}(h^c_i[:, k])$ is the softmax of the first $k$ channels of the final cell state, CE is the categorical cross entropy loss, where $t_i \in \{1, \ldots, k\}$ are the cell type targets. In practice we split the cross entropy loss of empty voxels and the cross entropy loss of the rest of voxels. This helps with training stability, as the loss for non-empty voxels does not overshadow the loss for the empty voxels. IOU is the Intersection Over Union loss which measures the absolute distance between non empty voxels and empty voxels. Given the loss the gradients are computed with back-propagation using automatic differentiation and the NCA parameters are trained with a gradient-based approach, using the Adam optimizer [51].

In this case the NCA quickly learns to grow the target structure, but is not stable when run for more time steps than it was trained for and can only recover from limited damage. In order to learn a NCA that is stable over thousands of time steps and capable of regenerating from severe damage we modify the training, following the approach outlined in [11].
Instead of always training from the same seed, at each training step we sample a batch of 5 random seeds from a pool of size 32. Of the 5 sampled seeds, we replace the one with the highest loss with a single cell seed, and apply damage to the two seeds with the lowest losses. We damage the seed by setting all the cell states to zero in a sphere with radius three, except the first channel, which corresponds to the empty block, which we set to one. After the training step we replace the sampled seeds in the pool with the output of the NCA at the last step, $h_t$. The pool is initialized with 32 single cell seeds. See Algorithm 1 for details.

\begin{algorithm}
\caption{NCA pool training with damage}
\begin{algorithmic}
\State $\text{pool} \leftarrow \text{init}()$
\While {not converged}
\State $\text{idx, seeds} \leftarrow \text{sample(pool, 5)}$
\State $\text{losses} \leftarrow \mathcal{L}(\text{seeds})$
\State $\text{seeds} \leftarrow \text{sort(seeds, losses)}$ \Comment{Sort seeds according to loss descending}
\State $\text{seeds}[0] = \text{initial}$ \Comment{Replace worst seed with initial seed}
\State $\text{seeds}[-2:] = \text{damage(seeds}[-2:])$ \Comment{Apply damage to two best seeds}
\State $T \sim \mathcal{U}[48, 64]$
\State $\text{seeds} \leftarrow \text{NCA(seeds, } T) $ \Comment{Run the NCA for $T$ steps}
\State $\text{loss} \leftarrow \sum \mathcal{L}(\text{seeds})$
\State $\text{optimize(NCA, loss)}$ \Comment{Train the NCA with gradient descent}
\State $\text{pool[idx]} \leftarrow \text{seeds}$ \Comment{Replace samples from pool with new seeds}
\EndWhile
\end{algorithmic}
\end{algorithm}

Fig. 7 Growing soft robots from a single cell with gradient-based trained NCAs.
4.2.2 Morphology similarity and locomotion recovery

We performed 20,000 steps of learning unless the loss function reaches 0. Training for 8903, 8066, and 20,000 steps for Biped, Tripod, and Multiped takes 15, 12, and 48 minutes using a GeForce GTX 1650 (TU117), respectively. The training curves for each morphology are shown in Fig. 6, which show a rapid decrease in the loss.

We first examined whether the trained network is able to grow the original target morphology from a single cell. The first single soft & passive cell is placed at
position \((3, 3, 3)\) with alive channel \(= 1\) and 10 hidden states with random numbers 0 to 1. All other cells are empty voxels with alive channel and 10 hidden states set to 0. After running NCA for 60 steps from the initial condition, we confirmed all the morphologies were grown and completely matched the original morphologies (Fig. 7).

Next, we evaluated the regeneration of morphology and the locomotion recovery by cutting half of the acquired original morphology. Based on the three spatial axes defined in Fig. 2, Left-Right, Top-Bottom, and Back-Front, six types of cuts were made to each of the three soft robots. We applied to change cell types to empty and set the active channel and hidden states to 0 for three of the seven columns of each axis. The cut-off morphologies were used as the initial state for the NCA and run for 200 steps for the Biped / Multipod and 150 steps for the Tripod. The NCA updates are shown in Fig. 8, and the similarity with the original morphology of the final state is shown in Table 2.

The Tripod was easier to regenerate than the other two morphologies and completely regenerated the original morphology for all damage, while the Biped only achieved 100% recovery for the top damage. However, it did not improve the similarity for the other types of damage even when run for more than 200 steps. Mutipod did not achieve 100% regeneration for any damage. In the first few steps (within 10) the similarity of the damaged robot to original is very high. However, to reach a perfect match, 100 steps are required.

We measured the locomotion of original, damaged, and regrown morphology with an evaluation time of 0.5s for 10 cycles in VoxCad. The travel distance for original morphology is shown in Table 1, and the rate of locomotion for damaged and regrown morphology is shown in Table 3. Biped’s left damage remained at 67% as in the evolutionary calculation (Table 1), while other damage critically lost its locomotion ability. In Tripod, all damaged morphology fatally lost movement ability but regained 100% movement ability as they completely regrew their original
morphology against all damage. In Multiped, the soft robot regained more than 80% of its mobility, except for Right and Top damage, where the soft robot collapsed in the middle due to the placement of a small number of cells and could not move any further, resulting in a small mobility value.

4.2.3 Comparing regeneration through evolutionary vs. differentiable programming

Comparing the percentage of similarity with the evolutionary training for Left damage, Biped was 99.4% compared to 98%, Tripod was 100% compared to 99%, and Multiped was 99.7% compared to 91%, with the differentiable training being higher for all three morphologies (Table 1 and 2). In addition, the similarity of all six damage types was greater than 95%, improving robustness to damage (Table 2).

Fig. 9 shows the locomotion of the original morphology, regrown morphology by evolutionary training, and regrown morphology by differentiable programming under the left damage condition using transparency 0.5 and transparency 0.5 with frame, respectively. For all three soft robots, the regrown morphology obtained through differentiable programming travelled further than that obtained by evolutionary optimization.

It is important to note that it is difficult to fairly compare these two different optimization approaches. Both methods not only use different optimization algorithms but also different neural network architectures and hyperparameters. However, while the efficiency of our genetic algorithm can likely be improved, the results reported here mirror similar results from previous research: artificial evolution often struggles when tasked to perform supervised learning [52] and even learning to grow simple 2D structures can be challenging [53]. In cases where a target structure is given, supervised learning through gradient-based optimization has shown to be the efficient method of choice, and has allowed significantly more complex 2D [11] and 3D patterns [10] to be grown through neural networks than have previously been possible. On the other hand, evolutionary algorithms have shown to be better suited for open-ended search [54] (instead of training towards a target) and the creative discovery of artefacts and behaviours [55]. The approach presented in this paper thus combines the advantages of both methods and we hope will spur the development of more of such hybrid methods.

5 Discussion and future work

The growth and regeneration ability of complex multicellular tissues are specific characteristics of biological systems that flexibly adapt to their environment. In order to apply this adaptive capability to artificial systems, in this paper, we gave the morphological regenerative ability to virtual soft robots. We have developed a new
method for soft robots to recover from damage based on neural cellular automata. The approach first evolves various growing soft robots and in a second steps trains them to being able to regenerate the original morphology against multiple damages through differentiable programming. Although complete regrowth to the original morphology was not always achieved, we confirmed that the soft robots regained more than 80% of travel distance in all damage conditions. These results suggest that growth can increase the evolutionary diversity of soft robot morphology and that regeneration can provide resilience to damage.

While the locomotion and regeneration tasks in this paper are relatively simple, they open up exciting future directions such as object manipulation, adaptation to environmental changes, task-based transformation, and self-replication.

Recently, soft robots designed using computer simulations have recently been recreated in real robots using a variety of materials [56]. With the development of material science, a variety of soft robots that can change their shape have also been created [57]. Currently, the technology of tissue culture has been developed, and hybrid robots with dynamic plasticity are being developed [36]. In the future, it may be possible to create a hybrid robot that can grow spontaneously and recovery function from damage by creating a soft robot designed using the proposed model with living tissue. Because the approach presented in this paper only relies on the local communication of cells, it could be a promising approach for the next generation of these hybrid robots.

Acknowledgements This work was supported by the Tobitate! (Leap for Tomorrow) Young Ambassador Program, a DFF-Research Project1 grant (9131-00042B), and KH’s Academist supporters1 (Takaaki Aoki, Hirohito M. Kondo, Takeshi Oura, Yusuke Kajimoto, Ryuta Aoki).

Funding S. Risi was funded by DFF-Starting Grant (Grant number 90063-00046B).

References

1. B.M. Carlson, *Principles of Regenerative Biology* (Elsevier/Academic Press, New York, 2011)
2. G.L. Wade, R.R. Westerfield, *Basic Principles of Pruning Woody Plants* (University of Georgia, 2009)
3. J.M. Davis, E.A. Estes, Spacing and pruning affect growth, yield, and economic returns of staked fresh-market tomatoes. J. Am. Soc. Hortic. Sci. 118(6), 719–725 (1993). https://doi.org/10.21273/JASHS.118.6.719
4. H.T. Hartmann, D.E. Kester et al., *Plant Propagation: Principles and Practices* (Prentice-Hall, New Jersey, 1975)
5. W.A. Vieira, K.M. Wells, C.D. McCusker, Advancements to the axolotl model for regeneration and aging. Gerontology 66(3), 212–222 (2020). https://doi.org/10.1159/000504294
6. M. Levin, J. Selberg, M. Rolandi, Endogenous bioelectrics in development, cancer, and regeneration: drugs and bioelectronic devices as electroceuticals for regenerative medicine. Iscience 22, 519–533 (2019). https://doi.org/10.1016/j.isci.2019.11.023
7. M.C. Vogg, B. Galliot, C.D. Tsiairis, Model systems for regeneration: Hydra. Development 146(21), 177212 (2019). https://doi.org/10.1242/dev.177212

1 https://academist-cf.com/projects/119?lang=en
8. N. Fausto, J.S. Campbell, K.I. Riehle, Liver regeneration. Hepatology 43(S1), 45–53 (2006). https://doi.org/10.1002/hep.20969
9. K. Horibe, K. Walker, S. Risi, Regenerating soft robots through neural cellular automata, in EuroGP, pp. 36–50 (2021)
10. S. Sudhakaran, D. Grbic, S. Li, A. Katona, E. Najarro, C. Glanois, S. Risi, Growing 3d artefacts and functional machines with neural cellular automata. arXiv preprint arXiv:2103.08737 (2021)
11. A. Mordvintsev, E. Randazzo, E. Niklasson, M. Levin, Growing neural cellular automata. Distill (2020). https://doi.org/10.23915/distill.00023.
12. K. Sims, Evolving 3d morphology and behavior by competition. Artif. Life 1(4), 353–372 (1994). https://doi.org/10.1162/artl.1994.1.4.353
13. K. Sims, Evolving virtual creaturesm, in Proceedings of the 21st Annual conference on computer graphics and interactive techniques, pp. 15–22 (1994). https://doi.org/10.1145/192161.192167
14. F. Dellaert, R.D. Beer, Co-evolving body and brain in autonomous agents using a developmental model. Cleveland, OH 44106 (1994)
15. P. Eggenberger, Evolving morphologies of simulated 3d organisms based on differential gene expression, in Proceedings of the Fourth European Conference on Artificial Life, pp. 205–213 (1997)
16. E.H. Ostergaard, H.H. Lund, Evolving control for modular robotic units, in Proceedings 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation. Computational Intelligence in Robotics and Automation for The New Millennium (Cat. No. 03EX694), vol. 2, pp. 886–892 (IEEE, 2003)
17. H. Lipson, J.B. Pollack, Automatic design and manufacture of robotic lifeforms. Nature 406(6799), 974–978 (2000). https://doi.org/10.1038/35023115
18. S. Risi, D. Cellucci, H. Lipson, Ribosomal robots: evolved designs inspired by protein folding, in Proceedings of the 15th Annual Conference on Genetic and Evolutionary Computation, pp. 263–270 (2013)
19. K.O. Stanley, Compositional pattern producing networks: a novel abstraction of development. Genet. Program Evolvable Mach. 8(2), 131–162 (2007). https://doi.org/10.1007/s10710-007-9028-8
20. N. Cheney, R. MacCurdy, J. Clune, H. Lipson, Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. ACM SIGEVOlution 7(1), 11–23 (2014). https://doi.org/10.1145/2661735.2661737
21. N. Cheney, J. Bongard, H. Lipson, Evolving soft robots in tight spaces, in Proceedings of the 2015 Annual Conference on Genetic and Evolutionary Computation, pp. 935–942 (2015)
22. N. Cheney, J. Bongard, V. SunSpiral, H. Lipson, Scalable co-optimization of morphology and control in embodied machines. J. R. Soc. Interface 15(143), 20170937 (2018). https://doi.org/10.1098/rsif.2017.0937
23. J.E. Auerbach, J.C. Bongard, Environmental influence on the evolution of morphological complexity in machines. PLoS Comput. Biol. 10(1), 1003399 (2014). https://doi.org/10.1371/journal.pcbi.1003399
24. J.E. Auerbach, J.C. Bongard, Evolving cppns to grow three-dimensional physical structures, in Proceedings of the 12th Annual Conference on Genetic and Evolutionary Computation, pp. 627–634 (2010)
25. J. Urzelai, D. Floreano, Evolutionary robotics: coping with environmental change, in Genetic and Evolutionary Computation Conference (GECCO 2000) (2000)
26. S. Nolfi, D. Floreano, Learning and evolution. Auton. Robot. 7(1), 89–113 (1999). https://doi.org/10.1023/A:1008973931182
27. K. Chatzilygeroudis, V. Vassiliades, J.-B. Mouret, Reset-free trial-and-error learning for robot damage recovery. Robot. Auton. Syst. 100, 236–250 (2018). https://doi.org/10.1016/j.robot.2017.11.010
28. A. Cully, J. Clune, D. Tarapore, J.-B. Mouret, Robots that can adapt like animals. Nature 521(7553), 503–507 (2015). https://doi.org/10.1038/nature14422
29. T. Kano, E. Sato, T. Ono, H. Aonuma, Y. Matsuzaka, A. Ishiguro, A brittle star-like robot capable of immediately adapting to unexpected physical damage. Royal Soc. Open Sci. 4(12), 171200 (2017). https://doi.org/10.1098/rsos.171200
30. E. Najarro, S. Risi, Meta-learning through hebbian plasticity in random networks. arXiv preprint arXiv:2007.02686 (2020)
31. S. Kriegman, N. Cheney, F. Corucci, J.C. Bongard, Interoceptive robustness through environment-mediated morphological development, in Proceedings of the Genetic and Evolutionary Computation Conference, pp. 109–116 (2018)
32. K. Walker, H. Hauser, Evolution of morphology through sculpting in a voxel based robot, in ALIFE 2021: The 2021 Conference on Artificial Life (MIT Press, 2021)
33. D.S. Shah, J.P. Powers, L.G. Tilton, S. Kriegman, J. Bongard, R. Kramer-Bottiglio, A soft robot that adapts to environments through shape change. Nat. Mach. Intell. 3(1), 51–59 (2021). https://doi.org/10.1038/s42256-020-00263-1
34. D. Shah, B. Yang, S. Kriegman, M. Levin, J. Bongard, R. Kramer-Bottiglio, Shape changing robots: bioinspiration, simulation, and physical realization. Adv. Mater. 33(19), 2002882 (2021). https://doi.org/10.1002/adma.202002882
35. S. Kriegman, S. Walker, D. Shah, M. Levin, R. Kramer-Bottiglio, J. Bongard, Automated shapeshifting for function recovery in damaged robots. arXiv preprint arXiv:1905.09264 (2019)
36. S. Kriegman, D. Blackston, M. Levin, J. Bongard, A scalable pipeline for designing reconfigurable organisms. Proc. Natl. Acad. Sci. 117(4), 1853–1859 (2020). https://doi.org/10.1073/pnas.1910837117
37. J. Von Neumann, in Theory of Self-Reproducing Automata, ed. by A.W. Burks (University of Illinois Press, 1966)
38. J.F. Miller, Evolving a self-repairing self-regulating french flag organism, in Proceeding of Genetic and Evolutionary Computation Conference, pp. 129–139 (Springer-Verlag, Berlin, 2004)
39. W. Gilpin, Cellular automata as convolutional neural networks. Phys. Rev. E 100(3), 032402 (2019). https://doi.org/10.1103/PhysRevE.100.032402
40. J.D. Hiller, H. Lipson, Multi material topological optimization of structures and mechanisms, in Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation, pp. 1521–1528 (2009)
41. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn Res 15(1), 1929–1958 (2014)
42. S. Hochreiter, Long short-term memory. Neural Comput. 1780, 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
43. M. Levin, G. Pezzulo, J.M. Finkelstein, Endogenous bioelectric signaling networks: exploiting voltage gradients for control of growth and form. Annu. Rev. Biomed. Eng. 19, 353–387 (2017). https://doi.org/10.1146/annurev-bioeng-071114-040647
44. K.A. McLaughlin, M. Levin, Bioelectric signaling in regeneration: Mechanisms of ionic controls of growth and form. Dev. Biol. 433(2), 177–189 (2018). https://doi.org/10.1016/j.ydbio.2017.08.032
45. J.H. Holland, Genetic algorithms. Sci. Am. 267(1), 66–73 (1992). http://www.jstor.org/stable/24939139
46. A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing. Nat. Comput. Ser. Springer, Berlin, Heidelberg (2003). https://doi.org/10.1007/978-3-662-05094-1
47. F.P. Such, V. Madhavan, E. Conti, J. Lehman, K.O. Stanley, J. Clune, Deep Neuroevolution: Genetic Algorithms are a competitive alternative for training deep neural networks for reinforcement learning. arXiv (2017) arXiv:1712.06567
48. S. Risi, K.O. Stanley, Deep neuroevolution of recurrent and discrete world models, in Proceedings of the Genetic and Evolutionary Computation Conference, pp. 456–462 (2019)
49. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
50. V. Nair, G.E. Hinton, Rectified linear units improve restricted Boltzmann machines, in International conference on machine learning (2010). https://icml.cc/Conferences/2010/papers/432.pdf
51. D.P. Kingma, J. Ba, Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
52. B.G. Woolley, K.O. Stanley, On the deleterious effects of a priori objectives on evolution and representation, in Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation, pp. 957–964 (2011)
53. S. Nichele, M.B. Ose, S. Risi, G. Tufte, CA-NEAT: evolved compositional pattern producing networks for celular automata morphogenesis and replication. IEEE Transact. Cognitive Develop. Syst. 10(3), 687–700 (2017). https://doi.org/10.1109/TCDS.2017.2737082
54. J.K. Pugh, L.B. Soros, K.O. Stanley, Quality diversity: A new frontier for evolutionary computation. Front. Robot. AI 3, 40 (2016). https://doi.org/10.3389/frobt.2016.00040
55. J. Lehman, J. Clune, D. Misievc, C. Adami, L. Altenberg, J. Beaulieu, P.J. Bentley, S. Bernard, G. Beslon, D.M. Bryson et al., The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities. Artif. Life 26(2), 274–306 (2020). https://doi.org/10.1162/artl_a_00319
56. T. Howison, S. Hauser, J. Hughes, F. Iida, Reality-assisted evolution of soft robots through large-scale physical experimentation: a review. arXiv (2020) arXiv:2009.13960

57. N. El-Atab, R.B. Mishra, F. Al-Modaf, L. Joharji, A.A. Alsharif, H. Alamoudi, M. Diaz, N. Qaiser, M.M. Hussain, Soft actuators for soft robotic applications: A review. Adv. Intell. Syst. 2(10), 2000128 (2020). https://doi.org/10.1002/aisy.202000128

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.