Exploring Material Design Space with a Deep-Learning Guided Genetic Algorithm

Kuan-Lin Chen
Department of Chemical and Biomolecular Engineering,
Johns Hopkins University, Baltimore, MD, USA

Rebecca Schulman
Department of Chemical and Biomolecular Engineering,
Johns Hopkins University, Baltimore, MD, USA
Department of Computer Science, Johns Hopkins University, Baltimore, MD, USA
Department of Chemistry, Johns Hopkins University, Baltimore, MD, USA

Abstract

Designing complex, dynamic yet multi-functional materials and devices is challenging because the design spaces for these materials have numerous interdependent and often conflicting constraints. Taking inspiration from advances in artificial intelligence and their applications in material discovery, we propose a computational method for designing metamorphic DNA-co-polymerized hydrogel structures. The method consists of a coarse-grained simulation and a deep learning-guided optimization system for exploring the immense design space of these structures. Here, we develop a simple numeric simulation of DNA-co-polymerized hydrogel shape change and seek to find designs for structured hydrogels that can fold into the shapes of different Arabic numerals in different actuation states. We train a convolutional neural network to classify and score the geometric outputs of the coarse-grained simulation to provide autonomous feedback for design optimization. We then construct a genetic algorithm that generates and selects large batches of material designs that compete with one another to evolve and converge on optimal objective-matching designs. We show that we are able to explore the large design space and learn important parameters and traits. We identify vital relationships between the material scale size and the range of shape change that can be achieved by individual domains and we elucidate trade-offs between different design parameters. Finally, we discover material designs capable of transforming into multiple different digits in different actuation states.

2012 ACM Subject Classification Computing methodologies → Artificial intelligence; Computing methodologies → Modeling and simulation

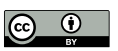
Keywords and phrases Machine Learning, Deep Learning, Computational Material Design, Multi-Objective Optimization, DNA Nanotechnology

Digital Object Identifier 10.4230/LIPIcs.DNA.2022.4

Supplementary Material *Software (Source Code)*: `https://github.com/charliecharlie29/Deep-Learning-Guided-Genetic-Algorithm`
archived at `swh:1:dir:b857ed74a5167e203a399609e9eb71bbad7cb091`

Funding This research is supported by Department of Energy under Grant No.DE-SC0010426 and by National Science Foundation under Grant No.EFRI-1830893 and by Army Research Office under Grant No.W911NF2010057.

Acknowledgements We would like to thank the Maryland Advanced Research Computing Center (MARCC) for providing cloud computing GPU services.



1 Introduction

Combinatorial Metamorphic Materials

The structure of a device determines its function. It is interesting to ask how we might manufacture *combinatorial metamorphic* devices that can take on many different forms and functions in response to a wide range of triggers. As metamorphic devices are capable of achieving a wide range of potential functions, they allow users to choose a function fit for a particular task and to alter the shape of the device to access that function. Because one such structure could be transformed into a large set of target devices, a single metamorphic structure can be stored or carried in place of a large set of traditional devices. These reasons make metamorphic devices more flexible and versatile than their traditional counterparts. These advantages have stimulated researchers from a range of communities to explore the design and construction of metamorphic devices[1, 3, 17, 21] for applications such as soft robotics[10, 21] and even quantum computing[5].

DNA-co-polymerized Hydrogels as Metamorphic Materials

Here we consider a means to construct metamorphic devices in which specific biochemical cues, DNA sequences, trigger the shape change of specific hydrogel domains[2]. Each hydrogel domain contains DNA crosslinks that allow the material to expand into swollen state upon activating the domain with a specific DNA sequence (growing actuators) and contract into shrunken state upon de-activation with another DNA trigger (shrinking actuators). To understand how we might use these types of materials to create metamorphic structures, we ask how to design material structures from 4 sets of the following active materials: hydrogels with a specific set of DNA crosslinks, growing actuators and shrinking actuators (Figure 1). Using multistage photolithography[2, 8, 18] to assemble these materials should allow us fabricate metamorphic devices that react to biomolecular signals with high specificity.

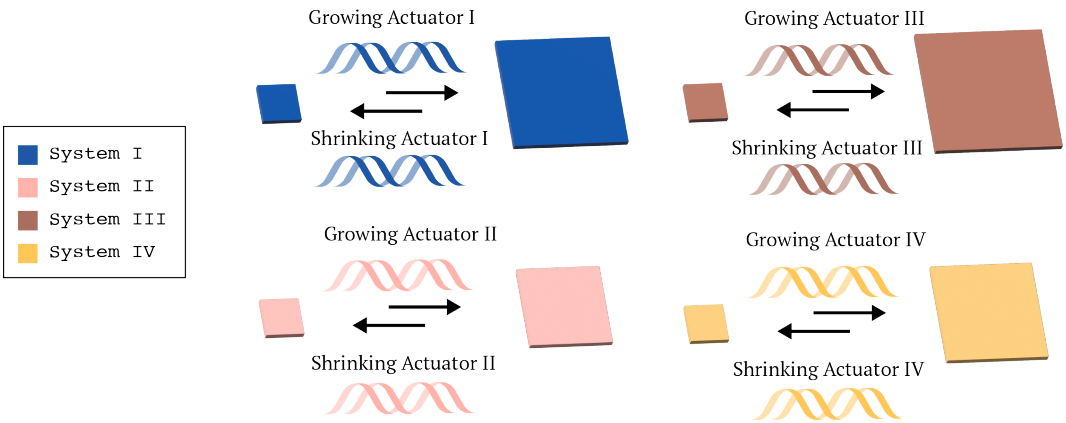


Figure 1 DNA-co-polymerized hydrogel. A set of 4 different materials that can be enlarged and contracted by DNA signal. Exposure to its shrinking signals prompts a material to enter its shrunken state, while exposure to its growing signal prompts a material to enter its swollen state. The signals are orthogonal (each affects only its target material and not others). Each material's expanded and shrunken states are of somewhat different sizes.

Metamorphic Multi-state Digit Transformer

DNA-co-polymerized hydrogels whose size can be bidirectionally controlled by DNA signals bring forth the possibility of building complex metamorphic materials. To explore this concept, we seek to investigate whether it would be possible to use the materials in Figure 1 to design a *metamorphic digit transformer*. We ask how one might build metamorphic devices using the 4 active materials with orthogonal DNA actuation systems and a passive material without a DNA actuation system. Each actuator system drives a specific domain into swollen state upon activation, and shrunken state when inactive. As a result, the device is multi-stable and has 16 possible states ($2^4 = 16$) when all DNA actuators are used. Our goal is to find a design where as many of its final outputs as possible resemble the shape of different digits from 0 to 9. We consider concatenated segments of bilayer hydrogel segments as a design space. The overall outline of this process is shown in Figure 2. Because these devices must be fabricated using multistage photolithography, the resulting designs must be consistent with this mode of fabrication, which imposes physical limits on the design space. Each bilayer segment will typically be on order 500μm in height and 250μm in width. In order for the curvature of the resulting structures to be governed by curvature along the lengthwise, segmented axis, the overall length of the structure must be significantly longer than either the structure's width or height. Lithography also limits the sizes of the segments to be between 50μm and 5000μm, below which size resolution becomes difficult to control, and above which would require a larger light source and mask. Moreover, we want to limit the number of distinct fabrication steps required, as difficulties such as alignment, device lift-off, and transfer increase rapidly with the number of fabrication steps.

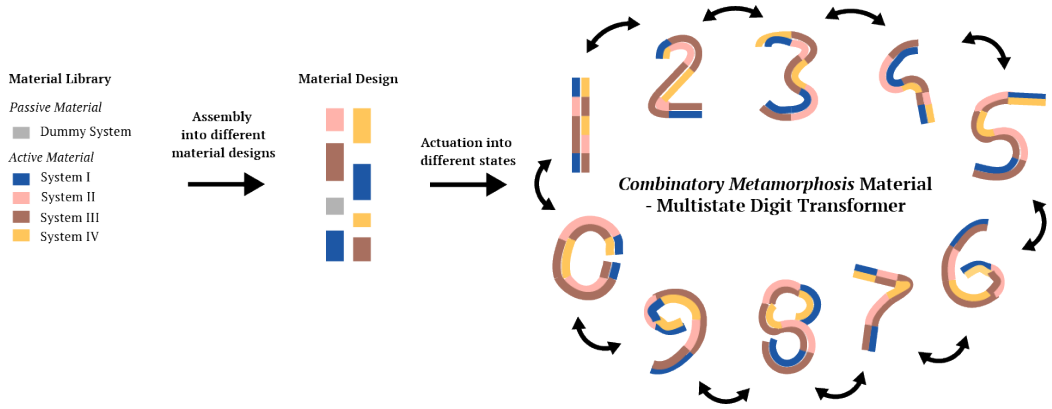


Figure 2 Building a metamorphic digit transformer from DNA-co-polymerized hydrogels. A metamorphic digit transformer is a stack of bilayers of DNA-co-polymerized hydrogels. The digit transformer would be able to change its shape into the shape of different digits upon the activation of different biochemical actuation programs, which switch the conformation of the structure between one of multiple stable states.

Machine Learning and Genetic Algorithm for Material Discovery

Finding a design for a structure with a large number of different forms is a challenge. This challenge increases as the number of design objectives (*i.e.* stable states) increases, and tighter physical and manufacturing limits are imposed. The design space, consisting of the types, arrangements, and lengths of hydrogel segments is too large to search through *via* trial and error, and the conflicting nature of the constraints also makes structured design methods challenging to consider. Inspired by how machine learning and artificial intelligence techniques have been transformative in different areas of material design and discovery[9, 11, 14, 20],

we seek to design digit transformer devices by developing a computational material discovery method. The method we describe mimics Darwinian evolution[4, 19] (through the use of a genetic algorithm) and combines numeric simulation[15] with state-of-the-art machine-learning models[16]. The resulting system simulates and autonomously evolves generations of material design variants *in-silico* to find designs for devices that satisfy the multiple design objectives[7] we created (Figure 2).

2 Methods

2.1 Material Simulation

To develop a geometric simulation platform for our DNA-co-polymerized hydrogel, we considered predicted values of changes in contour length (ΔL) and radii of curvature (RoC) during the expansion and contraction of bilayers consisting of different combinations of actuator types. The simulation of the material starts with defining a straight bilayer structured hydrogel with specified length and actuator pattern. We then look up the different values for RoC and ΔL given the target actuated state. The final folded shape is then calculated using the segment length, derived RoC and ΔL for each segment (Figure 3, top). We assume there is little stress along the horizontal axis and that the shape of the resulting structure is achieved by a linear "stack" of different segments then concatenate with one another to form a smooth curve (Figure 3, bottom).

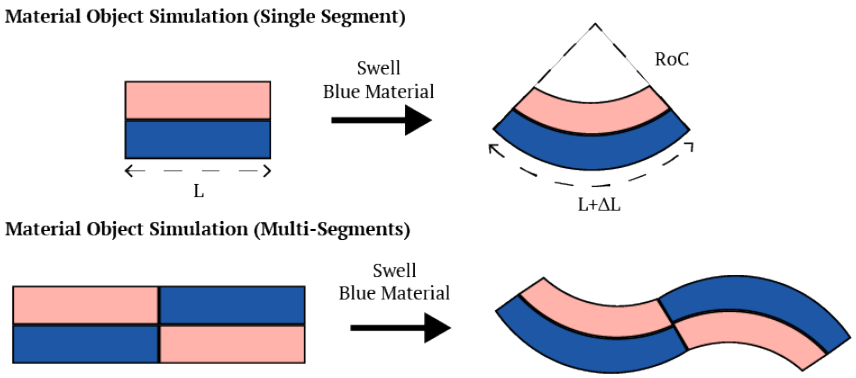


Figure 3 Simulation of DNA-co-polymerized bilayer hydrogel segment(s). The shape of a folded single segment is determined by the segment's length and the two types of actuators that make up the segment and their actuation states (expanded or contracted), as the states determine the overall values for the change in radius of curvature and contour length. Here, the bilayer studied has system I (blue) in its contracted state in the bottom segment and system II (pink) in its expanded state in the top segment. In a device with multiple segments, simulation is done with the assumption that there exists little stress between the segments so that the final conformation is the integrated sum of each single segment simulated independently.

2.2 Simulation of Device Curving

To construct a simulated "device," we create a list of segment lengths and a matrix of actuator types and their states. This design is encoded as a list S of segment lengths and a list P that encodes the actuator pattern (top then bottom) within each segment.

$$S = \{L_1, L_2, ..., L_n\} \tag{1}$$

$$P = \{\{X_{11}, X_{12}, ..., X_{1n}\}, \{X_{21}, X_{22}, ..., X_{2n}\}\} \tag{2}$$

where

$$X \in \{0, 1, 2, 3, 4\}$$

with 0 representing the passive system and 1 to 4 representing the 4 active materials.

A device's design is determined by simulating the curving of the device in all 16 possible states (where each of the four actuator types is in each of the two distinct states). An example of the possible states of a typical design is shown in Figure 4.

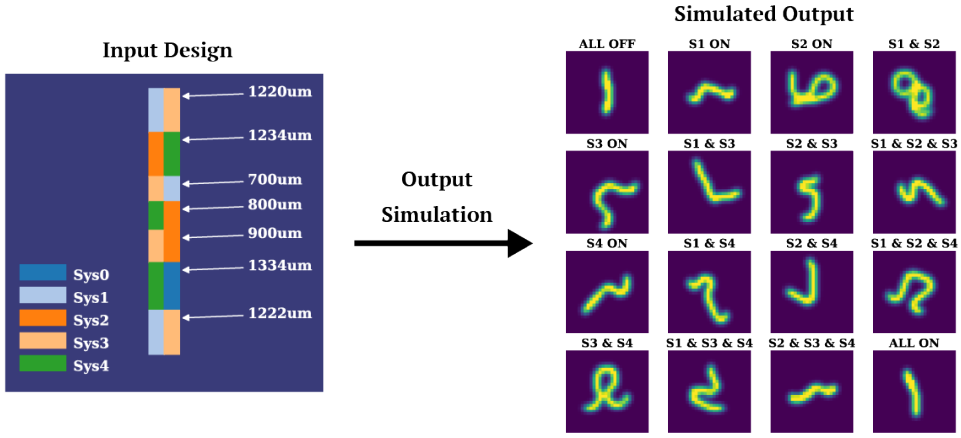


Figure 4 Example of a simulation of a device's curving in each of its 16 states. Shown are the input design and a map of the predicted output shapes. In the input section, different colors represent different actuators and the length of each bilayer segment is shown to the segment's right. The output diagrams show the predicted device shapes of all 16 possible actuation states. The label above each image specifies the actuator state (S2 designates system II, and so on, while ON designates the swollen state and OFF designates the shrunken state). For scoring purpose, the shapes are plotted in 28-by-28-pixels-images that are treated with a Gaussian blur filter ($\sigma = 1$).

2.3 Deep Learning Model for Design Selection

To efficiently automate the scoring and selection process for design optimization, we train a convolutional neural network (CNN) classifier[13] to distinguish the digit-similarity of the predicted geometric outputs. We use the *Tensorflow* library and train the model using a combinatory dataset consisting of the MNIST dataset[6] and an artificial dataset. We label twenty-four thousand images generated with the simulation platform manually to build an artificial dataset and add an additional class (class "10") for images that do not look digit-like and instead look like random squiggles. This additional class helps the model recognize bad shapes and images that the hydrogel device most often bends into. We train a sequential convolutional neural network consisting of two 2D convolutional layers (with 30 and 15 filters) with 2D max-pooling layers following each convolutional layer, and three fully connected layers (with 128, 50, and 11 nodes) and train on the combinatory data-set. The *relu* activation is used in all layers except for the final classification layer, where the *softmax* function is used. We used the *adam* optimizer[12] with categorical cross-entropy loss function and trained for 50 epochs. This model is then used to score and select ideal outputs that resemble digits. During the scoring and selection process, we rotate the images to explore the full potential of the designs.

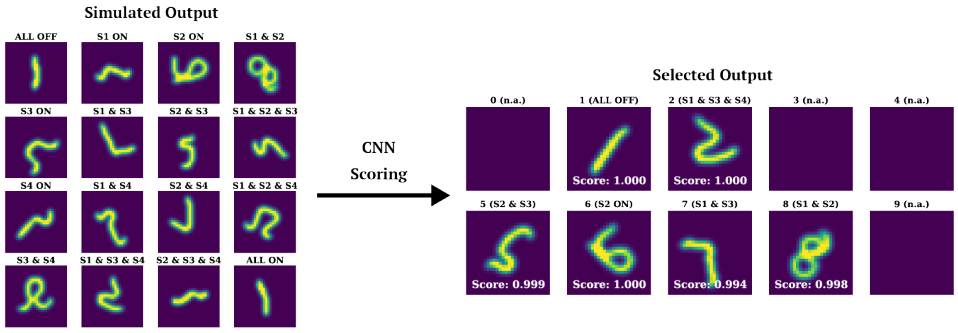


Figure 5 Example of design output scoring and selection. After the simulation, 16 images are generated and scored by the CNN to determine whether each represents a digit or not. Note that the final classification layer of the CNN model has 11 nodes where the first 10 represent the score (or probability) of resemblance to digit from 0 to 9. The 11-th value represents the value of resemblance to non-digit-like images. We use the max value of each class to represent the result of each image and discard the result if the 11-th value is the maximum - meaning the CNN model determines that this image is highly unlikely to represent any digits. Thus, only images with "max digit probability" are selected after the scoring and selection process of the CNN model.(Note that notations for simulated outputs are same as shown in Figure 4.)

2.4 Genetic Algorithm

We develop a genetic algorithm to evolve the material designs autonomously. The algorithm works by initiating a large batch of random designs with a fixed population size. At each iteration, the whole population is simulated and scored with the convolutional neural network(CNN) and a multi-objective loss function to determine the fitness score of each design.

2.4.1 Loss Function - Fitness Evaluation

We tried a variety of different loss function to track and optimize design:

Fitness Evaluation on Digit Quality Alone

We initially started the algorithm based on a simple loss function where only the digit quality is tracked and scored. This, however, leads to issues where designs that form a wide range of "mediocre digits" cannot compete with designs that form only a few number of "perfect digits", and we lose these designs throughout the evolution trajectory. While it is more likely for these "mediocre designs" to evolve and grow into designs that can fold into all digits, this loss function reduces the survival chance for them and thus are not ideal for the search.

$$fitness = \sum_{i=0}^{9} log(1 - V_i) \tag{3}$$

where

V_i: the score of each digits where $i \in \{0, 1, 2, ..., 10\}$

Fitness Evaluation on Digit Diversity and Quality

Moving forward, we improved the loss function so that we evaluate the performance based on the diversity of the digits formed as well as the quality of different digits. The diversity is evaluated with α, where we count how many digits a design formed. To calculate α for each design, we iterate through the classification results of the outputs and count how many of them are classified as digits (with *softmax*-ed classification value ranging from 0 to 9). α is the number of non-repeating digits formed. The quality of different digits is evaluated and stored in the list V, and $V_i = 0$ when the ith digit is not found. With this method we are more likely to find better designs and this loss function is used throughout the rest of the paper.

$$fitness = \sum_{i=0}^{9} log(1.0001 - V_i) \cdot \alpha \tag{4}$$

where:

V_i: the score of each digits where $i \in \{0, 1, 2, ..., 10\}$
α: diversity coefficient, the number of digits formed

2.4.2 Mutation Function

Once the whole population is scored, the population is then sorted according to the calculated fitness and 80% of the population is eliminated. The survivor designs are then delivered to a mutation function where we use the single-parent-mutation method to preserve the gene of each design and produce offspring. Each survivor design produces four offspring and sent with their offspring to the next generation to compete. This way the size of the population remains fixed throughout the evolution process. The iteration cycle continues until we reach the maximum generation limit.

Mutation Function - Vanilla Form

Mutation function is where we determine how the genetic information of the parent design is passed down to the offspring. The gene G in our designs consists of two matrices, the segment list S and the actuator pattern matrix P, such that $G = G(S, P)$. In the vanilla form of the mutation function, we randomly update either the S or P to mutate the genetic information, where each has a 50% chance of mutation. For the S-mutation offspring, we randomly assign new S while preserving the P. For the P-mutation offspring, we randomly swap out 50% of the genes within the pattern while preserving the S. Note that instead of swapping out all the genes, only 50% of them are mutated to ensure enough of the genetic information is maintained.

Mutation Function - With Fabrication Limit

Additionally, we use a more advanced mutation function that accounts for the fabrication complexity to ensure that the converging designs are within reasonable physical limits. The form of the function looks pretty similar to the vanilla form above except for additional fabrication step calculation. We define the fabrication steps as the steps needed to pattern these gels with the photolithography setup. The steps needed depends on the sum of different actuators on each side of the gel. For example, if we have a design that has actuator [1, 2, 1, 3, 1] on the one side and [4, 2, 2, 4, 2] on the other, it takes 3 steps to pattern the first

side given 3 different actuators used and 2 steps on the other. The total fabrication steps would be five to pattern these devices. The new mutation function now calculates the steps needed when the P is updated and rejects the P if it exceeds the maximum step allowed. The function would then reassign P and check and iterate until it converges on a new P that satisfies the fabrication limit.

Listing 1 Pseudo-code of the deep-learning guided genetic algorithm.

```
initialize first generation of designs

for i in range(generation_limit):
    fitness scoring with CNN model and loss function

    eliminate 80% of the population

    mutate survivors and generate descendants
```

Finally, we select the top 5 survivor designs to be the optimum converged designs in each evolution tree.

2.5 Search Evaluation

At the end of every evolution tree, the 5 final converged designs are saved and manually scored for an objective scoring; this is to mediate the possibility of false-positives from the convolutional neural net and to assure that the final outputs are digit-like to both machines and human. Figure 6 shows an example of the evaluation process, where each image gets a score of 1 (labeled with a green box) if it looks like a perfect digit, 0.5 (labeled with a blue box) if it looks similar but not perfect and 0 if unrecognizable. This is used as the final subjective-matrix to ensure the convergence of the model is reasonable.

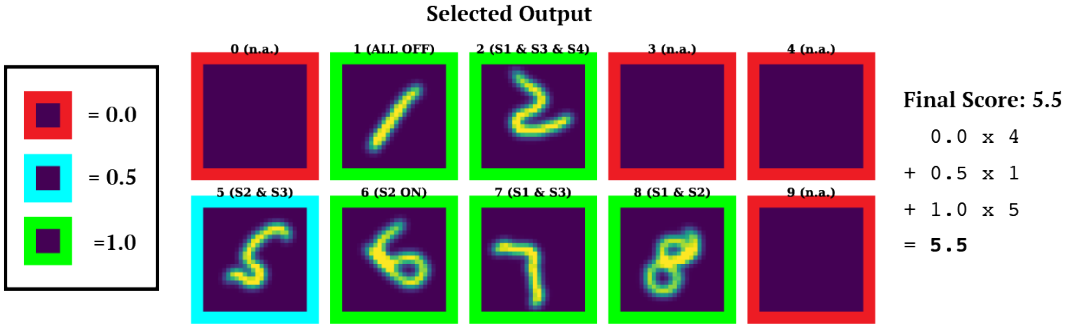


Figure 6 Example of manual(human-scored) evaluation. All converged outputs are manually scored to provide a final score for each design. In the example, the design forms 5 perfect digits (marked in green), 1 recognizable digit (marked in blue) and is unable to form other digits (marked in red). The final score is thus 1 x 5 + 0.5 x 1 = 5.5.

3 Results and Discussions

We then deployed the algorithm on the search for an optimal design. During the search, we seek to explore the parameter space and learn the effect of different hyper-parameters including - degree of freedom, design scale, search duration, and fabrication limit on the ability to find optimal design. As the searching process itself is stochastic and no independent event is repeatable, we evaluate the search of each condition with 3 separate evolution trees

and aggregate the information to avoid an independent event being an outlier. After the search, all converged designs are manually scored to assure the accuracy of evaluation. The distribution of final scores is used as a final metric, along with the loss trajectory, to help us determine whether a condition is ideal or not.

3.1 Degree of Freedom: Number of Segments

We first seek to learn the effect of how the number of segments affects the material's ability in finding digit transformer designs. During the search, we fixed the parameter of the total length to be roughly 8000μm and the search duration to be 100 generations. We ran the search of different segment lengths from 2 to 100 with 3 separate evolution trees for each condition. The result of the loss trajectory is shown in Figure 7 as an objective matrix to evaluate the effect of different segment numbers from the machine's perspective. The manually scored results of the converged designs of each condition are shown in Figure 8 as a subjective matrix for evaluation.

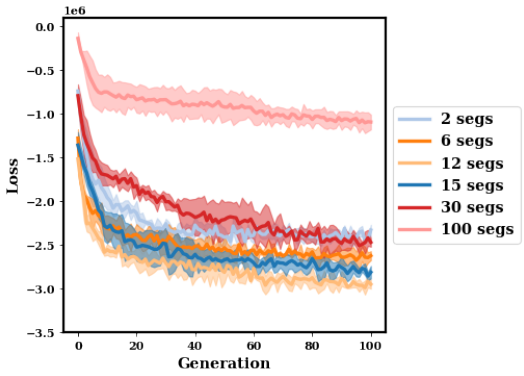


Figure 7 Loss trajectory of search on different segment number.

The number of segments is analogous to the degree of freedom of the system. More segments means there are more knobs to tune during the search and, theoretically speaking, better performance of the algorithm. The result, however, shows that this is only true within a certain extent. There exists a sweet-spot, and exceeding this region actually impales the ability to find good designs, as shown in Figure 8. We believe that this is because we are also increasing the complexity of the problem when we raise the number of segments used in the search, and after a certain amount of freedom is introduced, the benefit of having more knobs is out-weighted by the increase in complexity. Currently, our method and algorithm are unable to find good designs when the system is too complex. From this we learn the importance of maintaining the balance between degree of freedom and problem complexity when we are solving a problem with models.

3.2 Design Scale

Next we seek to learn the effect of the design scale on search performance. During the search, we fixed the parameter of number of segments used to 12 and the search duration to 100 generations. We ran the search of different total length scale from 800μm to 80,000μm with 3 separate evolution trees for each condition. The result of the loss trajectory is shown in Figure 9 as an objective matrix to evaluate the effect of different segment numbers from the machine's perspective. The manually scored results of the converged designs of each condition are shown in Figure 10 as a subjective matrix for evaluation.

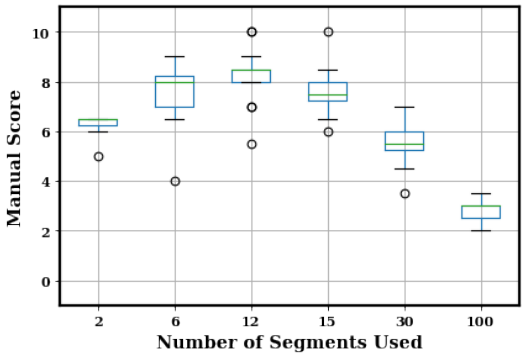


Figure 8 Score distribution of search on different segment number.

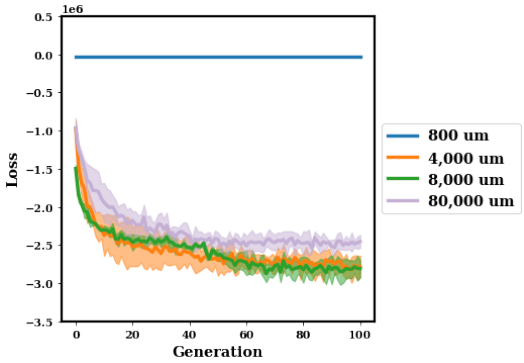


Figure 9 Loss trajectory of search on different length scale.

During the search, we learned that the length scale played an important role in the performance and that there also exists a sweet spot in the length scale when searching for an optimal design. We found out that when the length scale of the material is too small, the devices are unable to bend into large angles as the folding power the actuators provide is not enough. This limits the outputs of the devices to be simply straight or slightly bent 1s, and stops the algorithm from finding interesting designs. This is why the short length scale condition (800μm) only get score 1s in Figure 10, and we can see the algorithm is unable to optimize anything at all inspecting the loss trajectory in Figure 9. The search only starts becoming interesting and meaningful when the scale is large enough at 4,000μm, but the performance starts to decrease again when the scale becomes too large. This is because the devices are also more likely to misfold into undesired "random squiggles" when they are too long, which also corresponds to the folding power the actuators provide and the problem we are trying to solve. With this search we are able to locate the ideal length range for our devices given our actuator power and our target objectives.

3.3 Search Duration

Another parameter we changed during the search is the duration of the evolution. We fixed the parameter of number of segments used to be 12, the length scale to be 8,000μm, and search duration to be 150 generations with 3 separate evolution trees. We harvest and save the top 5 designs every 30 generations so we can track the manual scores at different evolution stages. The result of the loss trajectory is shown in Figure 11 as an objective matrix

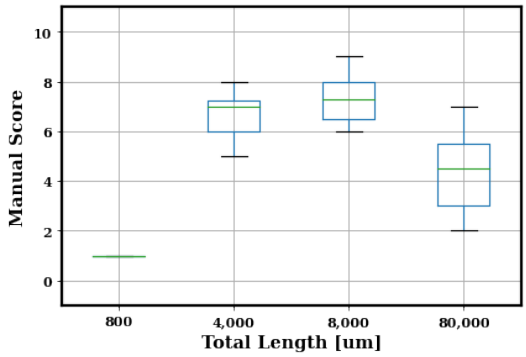


Figure 10 Score distribution of search on different length scale.

to evaluate the effect of different segment numbers from the machine's perspective. The manually scored results of the converged designs of each condition are shown in Figure 12 as a subjective matrix for evaluation.

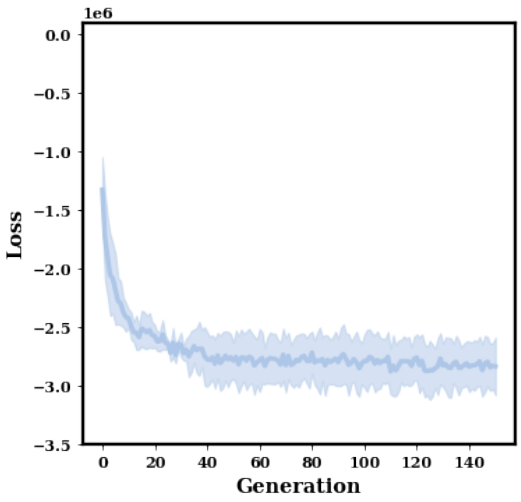


Figure 11 Loss trajectory of search on different search duration.

Here, we learn that designs start becoming meaningful quite early in the search, this makes sense as the algorithm is programmed to preserve all good designs found along the evolution path. The search also starts converging toward the minimum at around 90 to 120 generations, which is also why we set most search durations to 100 in other conditions. A more interesting point, however, is that we do not want the search to go on infinitely either. While the loss stopped decreasing after a certain amount of generations passed, we also observed a decrease in "gene diversity" when the search became too long. Here, we define "genes" $G(S, P)$ as the component that makes up the design input - the segment list S and pattern matrix P. We found out that as the search becomes too long, the genes especially in P start to become less diverse, with many survivors sharing similar P and the search process becoming a randomized S swapping search with little optimization going on. Moving forward, it may be helpful in future searches to add a gene diversity evaluation tracker within the search and program the mutation rate to change adaptively in response to the gene diversity.

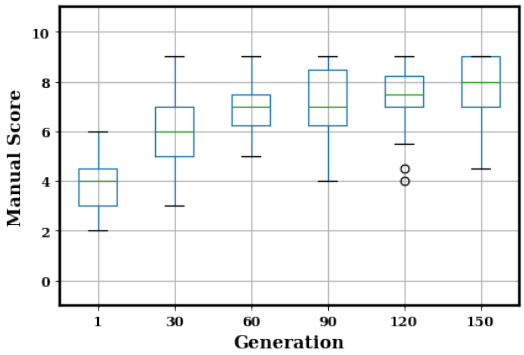


Figure 12 score distribution of search on different search duration.

3.4 Fabrication Limit

Finally, we investigated how fabrication limit affects the search performance, as it is also vital that the designs found should not exceed the physical patterning limit. Currently, we are imposing a 6 step limitation on the search, as exceeding this value increases the difficulty patterning the devices. During the search, we fixed the length scale to be 8,000μm and search duration to be 100 generations with 3 separate evolution trees for the two conditions - 6-segment designs and 12-segment designs. We compared the results with the same conditions shown above to learn how fabrication limit affects the search performance.

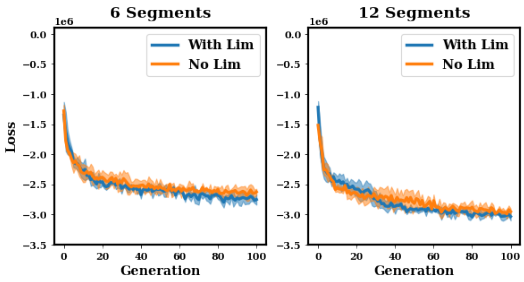


Figure 13 Loss trajectory of search on different fabrication limit.

We showed that the fabrication limit imposed does not drastically change the behavior or the performance of the search in our given conditions. While the limit does slightly decrease the performance of the optimal designs for the 12-segment search, we do think it is still possible to find a design that can fold into all digits given more evolution trees deployed for a wider and larger search. It is therefore encouraging that we now have a computational material discovery method that can autonomously learn and evolve the material designing process while considering real-world physical limitations.

4 Conclusion

Here, we demonstrate the development of a computational material discovery method for a multi-stable soft material with orthogonal actuators and automate the multi-objective optimization process with a genetic algorithm and integration of a deep-learning model. We show that we are able to efficiently explore the large parameter space and learn the effect of different variables. We also show that we can impose real-world physical constraints to discover reasonable designs.

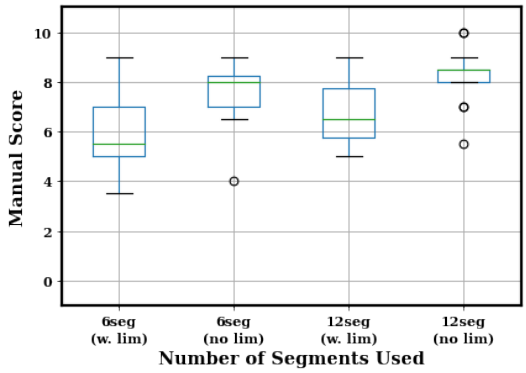


Figure 14 Score distribution of search on different fabrication limit.

In future work, one could expand the dimension of the simulation platform to handle more complex material actuation simulation. Building on this structure, it could also be possible to develop an advanced discovery platform that can optimize the functions of DNA-actuated hydrogel designs for tasks such as building locomotive robots.

References

1 Dhiraj Bhatia, Christian Wunder, and Ludger Johannes. Self-assembled, programmable dna nanodevices for biological and biomedical applications. *ChemBioChem*, 22(5):763–778, 2021.

2 Angelo Cangialosi, ChangKyu Yoon, Jiayu Liu, Qi Huang, Jingkai Guo, Thao D. Nguyen, David H. Gracias, and Rebecca Schulman. Dna sequence–directed shape change of photopatterned hydrogels via high-degree swelling. *Science*, 357(6356):1126–1130, 2017. doi:10.1126/science.aan3925.

3 VF Cardoso, C Ribeiro, and S Lanceros-Mendez. Metamorphic biomaterials. *Bioinspired Materials for Medical Applications*, pages 69–99, 2017.

4 N. Chakraborti. Genetic algorithms in materials design and processing. *International Materials Reviews*, 49(3-4):246–260, 2004. doi:10.1179/095066004225021909.

5 Peter W Deelman, Lisa F Edge, and Clayton A Jackson. Metamorphic materials for quantum computing. *MRS Bulletin*, 41(3):224–230, 2016.

6 Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

7 Abhijith M Gopakumar, Prasanna V Balachandran, Dezhen Xue, James E Gubernatis, and Turab Lookman. Multi-objective optimization for materials discovery via adaptive design. *Scientific reports*, 8(1):1–12, 2018.

8 Mustapha Jamal, Sachin S Kadam, Rui Xiao, Faraz Jivan, Tzia-Ming Onn, Rohan Fernandes, Thao D Nguyen, and David H Gracias. Bio-origami hydrogel scaffolds composed of photocrosslinked peg bilayers. *Advanced healthcare materials*, 2(8):1142–1150, 2013.

9 Paul C Jennings, Steen Lysgaard, Jens Strabo Hummelshøj, Tejs Vegge, and Thomas Bligaard. Genetic algorithms for computational materials discovery accelerated by machine learning. *NPJ Computational Materials*, 5(1):1–6, 2019.

10 Michał Joachimczak, Reiji Suzuki, and Takaya Arita. Artificial metamorphosis: Evolutionary design of transforming, soft-bodied robots. *Artificial life*, 22(3):271–298, 2016.

11 Yongfei Juan, Yongbing Dai, Yang Yang, and Jiao Zhang. Accelerating materials discovery using machine learning. *Journal of Materials Science & Technology*, 79:178–190, 2021. doi:10.1016/j.jmst.2020.12.010.

12 Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint*, 2014. arXiv:1412.6980.

13 Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. doi:10.1109/5.726791.

14 Yue Liu, Tianlu Zhao, Wangwei Ju, and Siqi Shi. Materials discovery and design using machine learning. *Journal of Materiomics*, 3(3):159–177, 2017. doi:10.1016/j.jmat.2017.08.002.

15 Arun Mannodi-Kanakkithodi and Maria KY Chan. Computational data-driven materials discovery. *Trends in Chemistry*, 3(2):79–82, 2021.

16 Tarak K. Patra, Venkatesh Meenakshisundaram, Jui-Hsiang Hung, and David S. Simmons. Neural-network-biased genetic algorithms for materials design: Evolutionary algorithms that learn. *ACS Combinatorial Science*, 19(2):96–107, 2017. doi:10.1021/acscombsci.6b00136.

17 Anjali Rajwar, Sumit Kharbanda, Arun Richard Chandrasekaran, Sharad Gupta, and Dhiraj Bhatia. Designer, programmable 3d dna nanodevices to probe biological systems. *ACS Applied Bio Materials*, 3(11):7265–7277, 2020.

18 Ruohong Shi, Joshua Fern, Weinan Xu, Sisi Jia, Qi Huang, Gayatri Pahapale, Rebecca Schulman, and David H Gracias. Multicomponent dna polymerization motor gels. *Small*, 16(37):2002946, 2020.

19 Changwon Suh, Clyde Fare, James A Warren, and Edward O Pyzer-Knapp. Evolving the materials genome: How machine learning is fueling the next generation of materials discovery. *Annual Review of Materials Research*, 50:1–25, 2020.

20 Rama Vasudevan, Ghanshyam Pilania, and Prasanna V Balachandran. Machine learning for materials design and discovery, 2021.

21 Zhi Zhao, Chao Wang, Hao Yan, and Yan Liu. Soft robotics programmed with double crosslinking dna hydrogels. *Advanced Functional Materials*, 29(45):1905911, 2019.