2023 American Control Conference (ACC)
San Diego, CA, USA. May 31 - June 2, 2023

Novelty Search for Neuroevolutionary Reinforcement Learning of Deceptive Systems: An Application to Control of Colloidal Self-assembly

Jared O'Leary, Mira M. Khare, Ali Mesbah

Abstract— **Colloidal self-assembly systems are generally difficult to control due to their highly nonlinear and stochastic dynamics and sparse rewards. These systems are also inherently deceptive, as successful control policies must be able to smooth out unavoidable defects and therefore temporarily move farther away from their goal in order to eventually realize the desired goal. This paper investigates the viability of evolutionary reinforcement learning (RL) based on *novelty search*, wherein behavioral novelty alone is used to learn control policies that can systematically mitigate deceptive dynamics. As such, for stochastic nonlinear systems that are prone to a deceptive behavior, novelty search is a promising alternative to the widely used *objective search* RL, where merely progress towards a pre-defined goal is used to learn and update control policies. In this work, we pair novelty search RL with a complexifying algorithm that simultaneously learns the neural network architecture and parameters of a control policy. The complexifying algorithm principles the novelty search by ensuring that simple behaviors must be discovered before more complex ones. We evaluate the performance of evolutionary RL based on objective search and novelty search on a benchmark *in-silico* colloidal self-assembly problem.**

I. Introduction

Colloidal self-assembly (SA) is the process by which discrete components (e.g., micro-/nano-particles in solution) spontaneously organize into an ordered state [1]. The spontaneous self-organization central to colloidal SA enables "bottom-up" materials synthesis, which can allow for manufacturing advanced, highly-ordered crystalline structures in an inherently parallelizable and cost-effective manner [2], [3]. The fact that colloidal SA can begin with micro- and/or nano-scale building blocks of varying complexity makes colloidal SA particularly suitable for bottom-up engineering and synthesis of metamaterials with unique optical, electrical, or mechanical properties [2], [3].

Nonetheless, colloidal SA is an inherently stochastic process prone to kinetic arrest due to particle Brownian motion [2]–[5]. This leads to variability in materials manufacturing and possibly high defect rates, which can severely compromise the viability of using colloidal SA to reproducibly manufacture advanced materials. This lack of reproducibility can in turn impede cost-effective and scalable manufacturing of these materials via SA processes [2], [3], [6], [7]. To this end, feedback control has shown promise for driving colloidal SA systems towards desired structures more reproducibly. In [8]–[10], proportional-integral control was demonstrated on simple test systems; however, such basic control algorithms may not be generalizable to colloidal SA systems with complex dynamics that are more prone to kinetic arrest. On the other hand, the control strategies proposed in [11]–[15] depend on learning stochastic dynamic models from data. These models generally take the form of the chemical Langevin equation, which may not be computationally tractable for real-time control of practically-sized systems with highly nonlinear dynamics [16].

Absent a stochastic model, [17]–[19] used reinforcement learning (RL) to learn control policies for colloidal SA. RL is a branch of machine learning concerned with learning to perform actions so as to achieve a desired objective (i.e., maximize a reward function). RL has been used extensively to manipulate robotics, enhance natural language processing, and outperform humans in video games, among many other applications [20]–[22]. Broadly, there exist two classes of RL methods: those that explicitly evaluate gradients of the reward function and those that do not [23]. Evolutionary RL methods fall under the latter class, as these methods propose and probabilistically accept changes to candidate control policies [24], [25]. Although less widely applied than gradient-based RL, evolutionary RL is naturally suited for "sparse-reward" problems such as colloidal SA, where the outcome of assembly (i.e., whether or not a desired structure forms) is not apparent until the later stages of the assembly process [18], [25], [26].

The evolutionary RL strategy in [18] uses artificial neural networks with thousands of parameters to represent the control policy for SA of chemically selective patchy discs. Despite the high complexity of the neural network representations, the resulting control policies had fairly simple (e.g., quadratic) relationships between the observed system states and inputs. Thus, a simpler control policy representation with fewer parameters might have yielded similar control performance. The simpler representation would require less training data and be more robust to overfitting. The apparent inconsistency between the complexity of the control policies' representation and behavior suggests that RL strategies that learn both the architecture and parameters of control policies simultaneously can be advantageous.

To our knowledge, all reported RL strategies for colloidal SA (e.g., those of [17]–[19]) learn control policies through *objective search*, where progress towards a pre-defined goal is measured and the control policies are iteratively updated according to this progress. Objective search RL, however,

Jared O'Leary, Mira Khare, and Ali Mesbah are with the Department of Chemical and Biomolecular Engineering, University of California, Berkeley, CA 94720, USA. {jared.oleary, mira_khare, mesbah}@berkeley.edu

Financial support from the National Science Foundation under Grant 2112754 is acknowledged.

can be prone to learning poor-performing control policies for "deceptive" systems that must first be guided farther away from their pre-defined goals before ever achieving them [27]. Complex colloidal SA systems are inherently deceptive, as successful control policies must be able to initiate disassembly of unavoidable kinetically arrested structures and then later initiate assembly to smooth out defects and achieve desired structures. Colloidal SA control policies must thus be able to temporarily move farther away from their goal in order to eventually achieve said goal [3], [12]. Although explored significantly less than objective search, RL methods based on *novelty search* can learn superior control laws for deceptive systems [27]. Novelty-search RL strategies search for behavioral novelty alone and create a cache of learned control policies that cause the "most novel" system behaviors. Then, the cached policy that best accomplishes a pre-defined goal is selected.

The objective of this work is to investigate the viability of evolutionary RL for learning the architecture and parameters of neural network control policies simultaneously while searching for behavioral novelty in learning such policies. In particular, we use a class of evolutionary RL methods called NEAT, or NeuroEvolution of Augmenting Topologies [28], to learn colloidal SA control policies via objective search and novelty search. NEAT learns artificial neural network (ANN) representations of control policies by evolving the network weights and architecture simultaneously. NEAT is initialized with ANNs that only contain input and output nodes. NEAT then evolves progressively larger ANN architectures. As a result, when paired with novelty search, the order in which new behaviors are discovered is principled (from less complex to more complex), rather than random [27]. In fact, NEAT with objective search and novelty search has been shown to learn effective control policies for a number of complex problems [27], [29]. However, to our knowledge, this study is the first application of novelty search to a system that is inherently deceptive and displays highly nonlinear and stochastic dynamics with multiplicative noise. To this end, the standard NEAT implementation is adapted to account for system stocasticity. We demonstrate NEAT with objective search and novelty search to learn closed-loop control policies that guide the colloidal SA of a benchmark *in-silico* system [11]. Open-source codes for these implementations are released on GitHub [30].

II. Problem Statement

Consider a stochastic nonlinear system

$$dx = f(x,u)dt + h(x,u)dw, \tag{1}$$

where x is the system state, u is the control input, t is time, w is a Gaussian white noise process, and $f(\cdot)$ and $h(\cdot)$ are nonlinear functions. As $h(\cdot)$ is a function of state and input, the noise is considered multiplicative. The input is constrained as $u \in \mathbb{U}$, where $\mathbb{U}$ is a compact set.

For the system (1), the goal of this work is to learn a control policy $u := \Pi(x;\theta)$ that maximizes some reward function $R(x)$. We represent $\Pi(x;\theta)$ by an ANN that is parameterized by its architecture and unknown weights and biases, collectively denoted by θ. We look to employ the evolutionary RL algorithm NEAT with objective search and novelty search to learn the "optimal" parameters θ^* that maximize the reward function $R(x)$

$$\theta^* := \arg\max_{\theta} \; R(x). \tag{2}$$

As such, θ^* yields the "optimal" parameterization of the control policy $\Pi(x;\theta^*)$, which is a function of the system state and must lie in $\mathbb{U}$.

III. Neuroevolution of Augmenting Topologies

NEAT, like most evolutionary RL algorithms, is initialized by creating a *population* of *genomes* [28]. At this stage, each genome is a candidate control policy that is represented by an ANN with randomly-selected weights and biases and the simplest possible architecture (e.g., no hidden nodes). During each *generation*, each genome's *fitness* is calculated. The genomes with the largest fitnesses are then chosen to be the "parents" of the next generation. These parents are probabilistically mutated, and the process repeats itself for a pre-determined number of generations, or until a pre-determined control performance is reached. Mutations can add nodes and node-to-node connections, change weight and bias values, or combine two parent genomes to create a new genome via "crossover" [28].

NEAT is unique in that it begins evolution with a population of small, simple ANNs and complexifies the network architecture over time. Although complexifying the architecture of an ANN does not always increase the complexity of the ANN's behavior, it does increase the upper bound of possible behavioral complexity by adding more parameters [20]–[22]. Simpler behaviors must thus be encountered before more complex behaviors [28]. A key feature distinguishing NEAT from prior work in complexification (e.g., [31], [32]) is its unique approach to maintaining a healthy diversity of ANN architectures. For example, NEAT uses the historical origins of genomes to inform which ones participate in crossover (i.e., two parent genomes "combining" to create a new genome). The use of this historical origin information allows NEAT to produce more meaningful offspring [28]. NEAT also employs speciation wherein only fitnesses among sufficiently similar architectures are compared during each generation. Speciation protects new structural innovations by reducing competition among differing architectures, thereby giving newer, more complex architectures room to adjust. Networks are assigned to species based on the extent to which they share historical origins. Complexification is thus supported by both historical origin and speciation, allowing NEAT to establish high-level features early in evolution and then elaborate on them later. As such, NEAT searches for a compact, appropriate network architecture by incrementally complexifying existing architectures.

IV. NEAT WITH OBJECTIVE SEARCH AND NOVELTY SEARCH

When NEAT is implemented with objective search, the fitness is calculated by evaluating the reward function $R(x)$. The goal of objective search is thus to find genomes that progressively yield larger reward function values. Objective-search RL, however, is prone to local optima and can perform poorly when applied to deceptive systems that must first be guided farther away from their pre-defined goals before ever achieving them [27], [29]. One reasonable explanation for this shortcoming is that the reward function does not necessarily reward the stepping stones in the search space that can ultimately lead to realizing the control objective (e.g., temporarily initiated disassembly to eventually achieve a defect-free crystal in colloidal SA). Novelty search, on the other hand, uses an alternative fitness criterion based on behavioral novelty alone. That is, instead of searching to maximize continuously the reward function, novelty search incentivizes genomes whose behavior is significantly different than what has been discovered before. Novelty-search RL thus calculates fitness based on a *novelty* metric that in no way measures overall progress.

For example, in the case of colloidal SA, initial candidate genomes may exclusively form weakly crystalline, amorphous structures. The novelty metric then rewards simply creating new structures, even if those structures are not close to the target, defect-free structure. In contrast, objective search may explicitly reward creating more crystalline structures, even if these structures are highly defective. When searching for novelty, after a few different weakly crystalline structures are discovered, the only way to be rewarded is to find a behavior that does not lead to a weakly crystalline structure. In this way, behavioral complexity arises from the bottom up. Eventually, to find a new behavior, candidate genomes would have to create a highly-ordered, defect-free structure, even if this structure is not explicitly included in the fitness calculation.

A natural question about novelty search is whether it follows any principle beyond naively enumerating all possible behaviors. Although novelty search does attempt to find all possible behaviors over time, when combined with a complexifying algorithm like NEAT, the order in which they are discovered is principled and not random. Recall that NEAT evolves increasingly complex neural networks. This way, the number of nodes and connections and, thus, the maximal complexity of neural networks discovered by novelty search increases over time, ensuring that simple behaviors must be discovered before more complex behaviors. Regardless of the particular encoding, this ordering from simple to complex is generally beneficial due to the minimum description-length principle in machine learning, i.e., the notion that the simplest satisfying description is usually the best [33].

A second natural question is whether novelty search is essentially identical to exhaustive search; enumerating all possible solutions will ultimately arrive at the best solution, but at an enormous computational cost. Many environments provide sufficient constraints on the types of behaviors that can actually be observed, without the need for further constraint from a reward function. For example, it is known that colloidal SA systems are prone to form kinetically arrested, highly-crystalline, yet highly defective structures [2], [3]. Although the control policy search-space is effectively infinite, the behavior-space into which points in the search-space collapse is limited, as systems often tend to become trapped in a handful of relatively similar configurations. In cases such as this, the search-space can collapse into a manageable number of novelty points, significantly differentiating novelty search from exhaustive search.

The novelty of a newly generated genome is computed with respect to the behaviors of previously evolved genomes, not their architectures. The aim is to characterize how far away the new genome is from its predecessors in the behavior-space. We measure sparseness and, thus, novelty with a k-nearest neighbors algorithm [34], where k is a fixed parameter that is determined empirically. As such, the sparseness $\rho(y)$ of behavior y is given by

$$\rho(y) = \frac{1}{k}\sum_{i=0}^{k} \text{dist}(y, \mu_i), \tag{3}$$

where μ_i is the ith nearest neighbor of y measured using the distance metric dist. In this work, we use the l_2-norm as the distance metric, and the "behavior" of y is value of the state variable x at various time points i. Areas with denser clusters of visited points are less novel and, therefore, rewarded less.

Algorithm 1 NEAT - The choice of fitness function determines whether the algorithm is based on novelty search (3), or objective search (5).

```
G ← Initpop(X, a)
best_predictions ← []
species_average ← []
for n in [1 . . . Y] do
    S ← speciate(G)
    for i in [1 . . . X] do
        C ← control(G(X), sys)
        F, P ← fitness(C), performance(C)
    best_predictions.append(G(max(P)))
    for j in [1 . . . len(S)] do
        species_average.append(mean(F(S(j))))
        if improvement(species_average(j)) is false then
            G ← remove(G(S(j)))
    mutate(max(P(S(j)))).append(G)
    if n = Y then
        Return best_predictions
P ← []
for n in [1 . . . M] do
    for i in [1 . . . len(best_predictions)] do
        C ← control(best_predictions(i), sys)
        P(i,n) ← performance(C)
Return G(max[mean(P(i,:))])
```

In Algorithm 1, NEAT with objective search and novelty search is summarized. X is the number of genomes, where each genome is an ANN with activation functions a. Initpop creates an initial population of genomes that only contain input and output nodes and no hidden nodes. Y is the number of generations. Speciate calculates the compatibility coefficient (a measure of how similar genomes are), and splits the population (G) into species. Control applies a control policy to the system. Fitness accesses the system's fitness, that is, (3) for novelty search, or (5) for objective search. Note that despite how fitness is written in the pseudocode, in our implementation, fitness is an average of the previous generation's fitness values and the current fitness for a genome that has existed for multiple generations. This averaging helps account for system stochasticity. Performance uses (5) to assess the control policy's performance. Improvement checks if the average species fitness has improved over some pre-determined number of generations. Mutate changes the genomes with the highest fitness values in each species by either adding a node, adding a node-node connection, changing a weight or bias value, or crossing over between 2 genomes in the species. M is the number of times the final set of control policies are tested on the system. This should be larger than 1 to minimize stochastic impacts on the final genome rankings.

V. Case Study: Colloidal Self-Assembly

A. System Description

We consider a low-dimensional, *in-silico* representation of a system of SiO_2 colloids from [11]. In this system, 174 identical SiO_2 particles with a nominal size of 1.5 μm are suspended in deionized water in a container made of glass microscope cover slips. Four separate, tunable 1 MHz AC electrode tips are attached to the edge of the container and are used to generate an electric field inside the container. The colloidal SA can be controlled by adjusting the voltage of the electric field. A discrete-time representation of this system is given by

$$\begin{aligned} x(i+1) &= g_1(x(i),u(i))\Delta t + \sqrt{2g_2(x(i),u(i))}w(i), \\ g_1(x,u) &= \frac{d}{dx}\Big(g_2(x,u)\Big) - \frac{d}{dx}\Big(F(x,u)\Big)\frac{g_2(x,u)}{K_bT}, \\ g_2(x,u) &= 4.5\times10^{-3}e^{-(x-2.1-0.75u)^2} + 0.5\times10^{-3}, \\ F(x,u) &= 10K_bT(x-2.1-0.75u)^2, \\ \Delta t &= 1 \text{ s}, T = 293 \text{ K}, K_b = 1.38066\times10^{-23} \text{ J/K}, \end{aligned} \quad (4)$$

where i is the time step, $w \sim N(0,1)$ is an independent and identically distributed Gaussian white noise term with zero mean and unit variance, $g_1(x,u)$ is the drift function, $g_2(x,u)$ is the diffusion landscape, and $F(x,u)$ is the free energy landscape. In this case, the state x is defined in terms of $<C_6>\in(0,6)$, where $<C_6>$ denotes the order parameter defined as the average number of hexagonally close packed particles around each particle, while u is the voltage of the applied external field. Note that $<C_6>= x \geq 5.0$ identifies a defect-free crystal [11]. Note that while (4) is a discrete-time representation, Algorithm 1 can be applied to continuous-time systems as well.

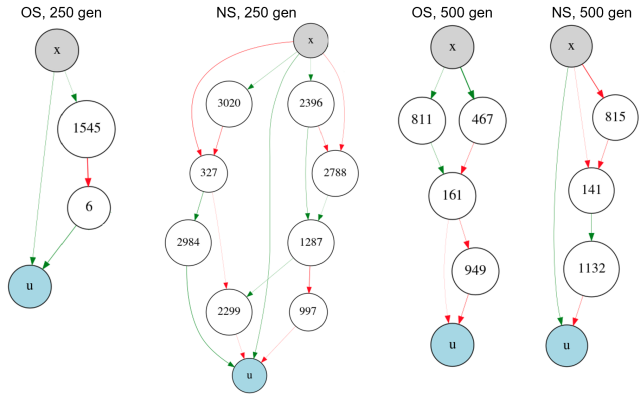

Fig. 1: The artificial neural network architectures for the control policies from Fig. 2 that were trained with 250 and 500 generations."NS" denotes novelty search, "OS" objective search, and "gen" the number of generations used to train a given control policy. The numbers in each node are historical markings used by NEAT.

The control objective is to learn control policies (i.e., functions $\Pi(x;\theta^*)$ that map the system state x at time i to some control input u) that manipulate the SA of the SiO_2 particles into defect-free, two-dimensional hexagonal crystals (i.e., maximize x) as quickly as possible over some pre-determined batch time, N. We thus seek to use NEAT with objective search and novelty search to learn control policies that maximize the reward function

$$R(x) = -\sum_{i=0}^{N}(x - x_{\max})^2, \quad (5)$$

where $x_{\max} = 6.0$, i.e., the maximum physical value of x.

B. Closed-Loop Implementations

Algorithm 1 with objective search and novelty search was implemented on the colloidal SA system. Each NEAT trial used 150 genomes and either 125, 250, or 500 generations. Note that, in this work, the implementation of NEAT is unique in that it explicitly accounts for run-to-run variability by applying select genomes to the system multiple times and then averaging the calculated fitnesses (see the end-to-end

TABLE I: Relative performance of NEAT with objective search and novelty search in relation to the number of generations. The performance is evaluated in terms of (5) for both cases. All performances are scaled to the lowest possible reward function value. The lowest performing algorithm would have a value of 1.0.

	Number of Generations		
	125	250	500
Novelty Search	1.04	1.13	1.16
Objective Search	1.0	1.04	1.09

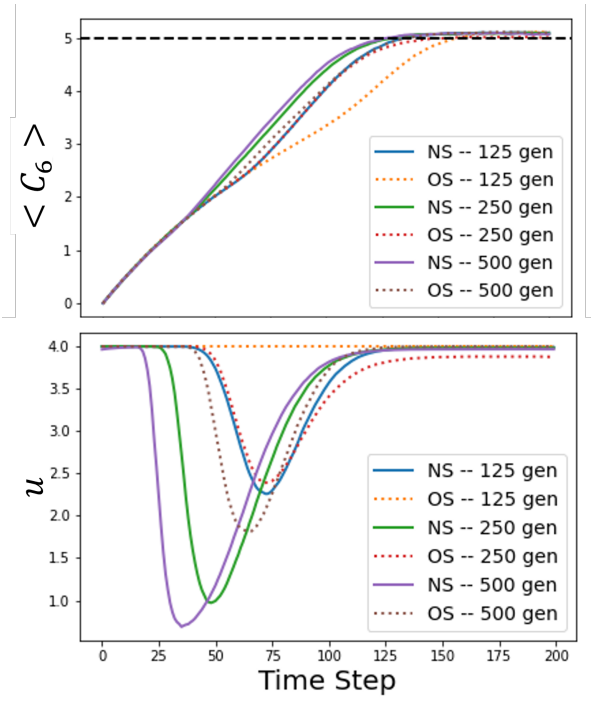

Fig. 2: State and input trajectories (averaged over 1000 replicates) for a few representative control policies. "NS" denotes novelty search, "OS" objective search, and "gen" the number of generations used to train a given control policy. Policies learned with novelty search create target crystals more quickly and have more complex input profiles than those learned with objective search.

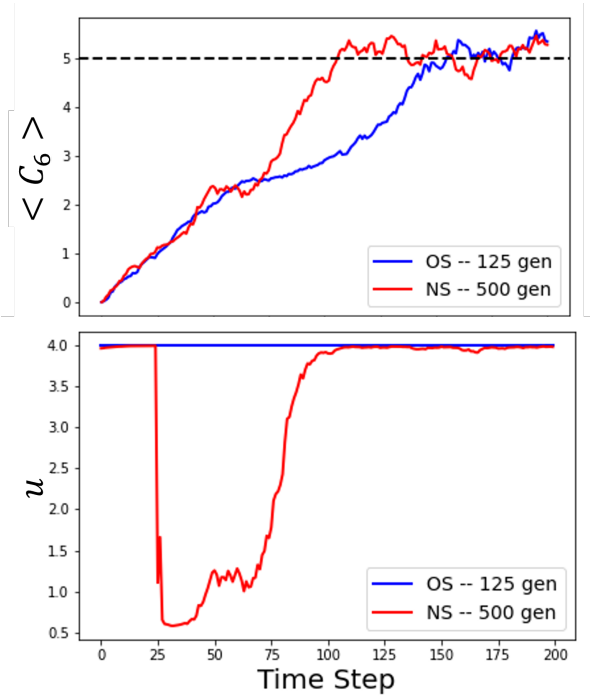

Fig. 3: State and input trajectories for the lowest- and highest-performing control policies from Fig. 2. "NS" denotes novelty search, "OS" objective search, and "gen" the number of generations used to train a given control policy.

implementation in [30] for more details). The "behavior" y used to calculate novelty in (3) is defined as as a ten-dimensional vector since $x(i)$ is recorded at every ten steps within the batch time N. In this case, smaller and larger behavior vectors showed no meaningful difference in the performance of the learned control policies.

Fig. 1 shows the architectures of the ANN representations of 4 control policies whose average behavior is depicted in Fig. 2. At 250 generations, novelty search yields both a more complex architecture and more complex behavior than objective search. Interestingly, the architectures of control policies from both search strategies at 500 generations are nearly identical despite the policies encoding behaviors of different complexities. This apparent inconsistency in architecture and behavioral complexity is not unexpected, as ANN architecture merely limits the upper bound of possible behavioral complexity. Note how simple the architectures of the learned policies are despite encoding relatively complex behaviors. Meanwhile, the ANNs used to control colloidal SA in [18] encoded less complex behavior while containing thousands of parameters. The relative simplicity of the ANN architectures in Fig. 1 suggests that future colloidal SA RL strategies may focus on smaller architectures, which may be more robust to over-fitting and easier to learn.

Table I shows that control policies learned from novelty search consistently outperform those learned from objective search. For both objective search and novelty search, control policy performance increases with the number of generations (i.e., the number of updates to the control policy); however, novelty search outperforms objective search at each generation number. The ability of NEAT with novelty search to learn effective control policies with relatively few generations supports the earlier hypothesis that the behavior-space can collapse into a manageable number of novelty points, which would significantly differentiate novelty search from exhaustive search.

Fig. 2 shows the averaged state and input trajectories for six representative control policies. Although both objective search and novelty search consistently yield defect-free crystals well before the end of the batch time, the novelty search control policies assemble these high-quality crystals more quickly and, thus, yield larger values of the reward function (5). At first glance, Fig. 2 shows that novelty search leads to more complex (i.e., nonlinear) control policies. A closer look reveals that novelty search control policies account for deception more effectively than objective search policies. For example, an input value of $u = 4.0$ most strongly initiates assembly. When u is decreased, the driving force towards assembly is also decreased. The 125-generation, objective search control policy holds the input at $u = 4.0$. Meanwhile, the 500-generation novelty search control policy brings the control input close to its physical minimum of $u = 0.5$ before slowly initiating assembly more strongly.

Regardless of generation number, the novelty search control policies consistently reach lower levels of u and outperform the objective search control policies as a result. We note, however, that objective search control policies more effectively mitigate deception and perform better as the generation number is increased.

Fig. 3 shows how NEAT with novelty search can yield control policies that mitigate deception. The figure shows example single trajectories from the lowest- and highest-performing control policies from Fig. 2, that is, objective search with 125 generations and novelty search with 500 generations, respectively. The objective search policy holds the electric field voltage u constant at its maximum physical value for the entire trajectory. This constant voltage always initiates assembly regardless of the state. Meanwhile, the novelty search policy oscillates u to temporary initiate disassembly (e.g., time steps 50-60) to assemble a defect-free crystal more quickly than holding u constant would.

VI. Conclusions and Future Work

The paper investigates the viability of neuroevolutionary RL with objective search and novelty search for mitigating deception within and, thus, effectively controlling stochastic nonlinear systems. Objective search and novelty search were paired with the NEAT algorithm, which learns ANN representations of control policies by evolving network weights and architectures simultaneously. In this way, NEAT discovers simpler behaviors before more complex ones and, thus, principles novelty search. Our results on a benchmark colloidal self-assembly system showed that novelty search NEAT consistently outperforms objective search. However, NEAT is limited in how complex of control policies it can learn. Although we argue that control of colloidal SA may not in fact necessitate large ANN architectures, certain systems may require ANN control policy architectures that contain hundreds of nodes and tens of layers. As such, a focus of future work will be applying novelty search to RL strategies that involve complex architectures. We will also explore the use of novelty search with gradient-based RL.

References

[1] G. M. Whitesides and B. Grzybowski, "Self-assembly at all scales," *Science*, vol. 295, no. 5564, pp. 2418–2421, 2002.

[2] J. A. Paulson, A. Mesbah, X. Zhu, M. C. Molaro, and R. D. Braatz, "Control of self-assembly in micro-and nano-scale systems," *Journal of Process Control*, vol. 27, pp. 38–49, 2015.

[3] X. Tang and M. A. Grover, "Control of microparticle assembly," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 5, pp. 491–514, 2022.

[4] D. T. Gillespie *et al.*, "Stochastic simulation of chemical kinetics," *Annual Review of Physical Chemistry*, vol. 58, no. 1, pp. 35–55, 2007.

[5] D. T. Gillespie, A. Hellander, and L. R. Petzold, "Perspective: Stochastic algorithms for chemical kinetics," *The Journal of Chemical Physics*, vol. 138, no. 17, p. 05B201_1, 2013.

[6] E. M. Furst, "Directed self-assembly," *Soft Matter*, vol. 9, no. 38, pp. 9039–9045, 2013.

[7] J. A. Liddle and G. M. Gallatin, "Nanomanufacturing: a perspective," *ACS Nano*, vol. 10, no. 3, pp. 2995–3014, 2016.

[8] J. J. Juárez and M. A. Bevan, "Feedback controlled colloidal self-assembly," *Advanced Functional Materials*, vol. 22, no. 18, pp. 3833–3839, 2012.

[9] Y. Gao and R. Lakerveld, "Feedback control for defect-free alignment of colloidal particles," *Lab on a Chip*, vol. 18, no. 14, pp. 2099–2110, 2018.

[10] Y. Gao and R. Lakerveld, "Feedback control for shaping density distributions of colloidal particles in microfluidic devices," *Lab on a Chip*, vol. 19, no. 13, pp. 2168–2177, 2019.

[11] X. Tang, Y. Xue, and M. A. Grover, "Colloidal self-assembly with model predictive control," in *Proceedings of the American Control Conference*, 2013, pp. 4228–4233.

[12] X. Tang, B. Rupp, Y. Yang, T. D. Edwards, M. A. Grover, and M. A. Bevan, "Optimal feedback controlled assembly of perfect crystals," *ACS Nano*, vol. 10, no. 7, pp. 6791–6798, 2016.

[13] X. Tang, J. Zhang, M. A. Bevan, and M. A. Grover, "A comparison of open-loop and closed-loop strategies in colloidal self-assembly," *Journal of Process Control*, vol. 60, pp. 141–151, 2017.

[14] I. Nodozi, J. O'Leary, A. Mesbah, and A. Halder, "A physics-informed deep learning approach for minimum effort stochastic control of colloidal self-assembly," *arXiv preprint arXiv:2208.09182*, 2022.

[15] M. A. Bevan, D. M. Ford, M. A. Grover, B. Shapiro, D. Maroudas, Y. Yang, R. Thyagarajan, X. Tang, and R. M. Sehgal, "Controlling assembly of colloidal particles into structured objects: Basic strategy and a case study," *Journal of Process Control*, vol. 27, pp. 64–75, 2015.

[16] J. O'Leary, J. A. Paulson, and A. Mesbah, "Stochastic physics-informed neural ordinary differential equations," *Journal of Computational Physics*, vol. 468, p. 111466, 2022.

[17] J. Zhang, J. Yang, Y. Zhang, and M. A. Bevan, "Controlling colloidal crystals via morphing energy landscapes and reinforcement learning," *Science Advances*, vol. 6, no. 48, p. eabd6716, 2020.

[18] S. Whitelam and I. Tamblyn, "Learning to grow: Control of material self-assembly using evolutionary reinforcement learning," *Physical Review E*, vol. 101, no. 5, p. 052604, 2020.

[19] S. Whitelam and I. Tamblyn, "Neuroevolutionary learning of particles and protocols for self-assembly," *Physical Review Letters*, vol. 127, no. 1, p. 018003, 2021.

[20] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[21] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017.

[22] P. Dayan and Y. Niv, "Reinforcement learning: the good, the bad and the ugly," *Current Opinion in Neurobiology*, vol. 18, no. 2, pp. 185–196, 2008.

[23] S. Whitelam, V. Selin, S.-W. Park, and I. Tamblyn, "Correspondence between neuroevolution and gradient descent," *Nature Communications*, vol. 12, no. 1, pp. 1–10, 2021.

[24] D. E. Moriarty, A. C. Schultz, and J. J. Grefenstette, "Evolutionary algorithms for reinforcement learning," *Journal of Artificial Intelligence Research*, vol. 11, pp. 241–276, 1999.

[25] S. Whiteson, "Evolutionary computation for reinforcement learning," *Reinforcement Learning*, pp. 325–355, 2012.

[26] S. Khadka and K. Tumer, "Evolution-guided policy gradient in reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[27] J. Lehman and K. O. Stanley, "Abandoning objectives: Evolution through the search for novelty alone," *Evolutionary Computation*, vol. 19, no. 2, pp. 189–223, 2011.

[28] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," *Evolutionary Computation*, vol. 10, no. 2, pp. 99–127, 2002.

[29] J. Lehman, "Evolution through the search for novelty," *PhD Dissertation, University of Central Florida*, 2012.

[30] J. O'Leary, "NEAT with objective search and novelty search for controlling colloidal self-assembly," https://github.com/jtoleary/csa_neat_ns, online; accessed 1 August 2022.

[31] F. Gruau, D. Whitley, and L. Pyeatt, "A comparison between cellular encoding and direct encoding for genetic neural networks," in *Proceedings of the 1st Annual Conference on Genetic Programming*, 1996, pp. 81–89.

[32] X. Yao, "Evolving artificial neural networks," *Proceedings of the IEEE*, vol. 87, no. 9, pp. 1423–1447, 1999.

[33] P. D. Grünwald, I. J. Myung, and M. A. Pitt, *Advances in minimum description length: Theory and applications*. MIT press, 2005.

[34] O. Kramer, "K-nearest neighbors," in *Dimensionality Reduction with Unsupervised Nearest Neighbors*. Springer, 2013, pp. 13–23.