

# Bayesian Particles on Cyclic Graphs

Ana Pervan<sup>1</sup> and Todd D. Murphey<sup>1</sup>

**Abstract**—We consider the problem of designing synthetic cells to achieve a complex goal (e.g., mimicking the immune system by seeking invaders) in a complex environment (e.g., the circulatory system), where they might have to change their control policy, communicate with each other, and deal with stochasticity including false positives and negatives—all with minimal capabilities and only a few bits of memory.

We simulate the immune response using cyclic, maze-like environments and use targets at unknown locations to represent invading cells. Using only a few bits of memory, the synthetic cells are programmed to perform a reinforcement learning-type algorithm with which they update their control policy based on randomized encounters with other cells. As the synthetic cells work together to find the target, their interactions as an ensemble function as a physical implementation of a Bayesian update. That is, the particles act as a particle filter.

This result provides formal properties about the behavior of the synthetic cell ensemble that can be used to ensure robustness and safety. This method of simplified reinforcement learning is evaluated in simulations, and applied to an actual model of the human circulatory system.

## I. INTRODUCTION

As robot size decreases to the order of a single cell, previously inconceivable applications and abilities emerge. These include monitoring of oil and gas conduits [1], electrophysiological recordings with neural dust motes [2], minimally invasive medical procedures [3], and much more. In this work, we investigate the use of synthetic cells to imitate some of the functionality seen in the immune system.

The immune system protects the body by recognizing and responding to antigens, which are harmful agents like viruses, bacteria, and toxins [4]. When white blood cells find a target, they multiply and send signals to other cells to communicate their discovery [5]. We show that a group of synthetic cells can imitate this discovery and communication behavior, by collectively executing a type of reinforcement learning that manifests itself as a Bayesian update over the control policy that brings cells to the location of an antigen.

Synthetic cells are microscopic devices with limited sensing, control, and computational capabilities [6]. They can contain simple circuits that include minimal sensors and very limited nonvolatile memory—barely a handful of bits [7]. These devices are around  $100\mu\text{m}$  in size, rendering classical computation using a CPU impossible. But simple movement, sensory, and memory elements can potentially be combined with a series of physically realizable logical operators to enable a specific task [8], [9] (e.g., a simple reinforcement

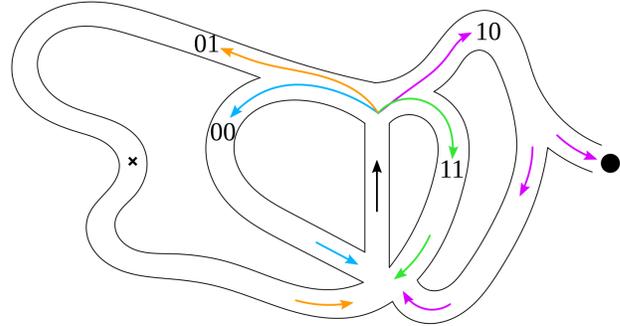


Fig. 1: A cyclic maze that mimics aspects of the circulatory system. A group of synthetic cells would require at least three bits each to navigate through the four paths and record instances of target detection. One of the paths (10) leads to a juncture where cells might get lost and not return.

learning algorithm for finding a target) and communication about how to achieve that task.

Reinforcement learning is centered on finding a suitable action to maximize a reward in a particular situation [10]. Unlike in supervised learning, where an agent is trained using examples of optimal outputs, in reinforcement learning an agent must choose its actions by learning through experience. This is especially important in applications related to medicine and biology, where no two situations are the same. Robots cannot be preprogrammed to perfectly achieve a task in such an environment—we can expect that they must have some element of online learning, and in the context of synthetic cells the question is how they can learn without traditional computation.

A version of reinforcement learning can be executed in a group of synthetic cells that begin with different control policies—indicating *how*, and implicitly where, they should explore—and then communicate with each other that they have or have not been successful in detecting a target. After communicating their success, some synthetic cells will change their control policies to reflect the successes of others in the group. Thus the distribution of synthetic cell control policies reflects the expected location of the target.

This update is similar to a particle filter, where samples from a distribution are represented by a set of particles, and each particle has a likelihood weight assigned to it that corresponds to the probability of that particle being sampled from the distribution. Particle filters also include a resampling step, to mitigate weight disparity before the weights become too uneven, which closely mirrors the communication step in this synthetic cell implementation, as we discuss in Section IV.

In this paper we show how synthetic cells can use simple, local algorithms and only a few bits of memory to enable global reinforcement learning behavior to refine their belief

<sup>1</sup>Department of Mechanical Engineering, Northwestern University, Evanston, Illinois, USA anapervan@u.northwestern.edu t-murphey@northwestern.edu

of a target location. We also show that this implementation of the ensemble of synthetic cells is a suboptimal Bayesian filter, where the control policy of the cells is the decision variable. By constraining synthetic cells to behave as a Bayesian update, the group of cells inherits formal properties in the form of guarantees on asymptotic performance and probabilistically predictable behavior. These properties will help us to reason about robustness and safety in task execution. That is, we are replacing the model of a distributed system with a single Bayesian filter.

## II. RELATED WORK

Literature surveys of previous work on nanotechnology and mobile microrobots can be found in [7] and [3], respectively. In the discussion of existing challenges associated with designing miniaturized robots for biomedical applications, [3] notes that most robots with dimensions less than  $1mm$  use an “off-board” approach where the devices are externally actuated, sensed, controlled, or powered. In this work, we employ fully autonomous devices that process information and act independently of external drivers and centralized computers.

Research in nanofabrication and synthesis methods have yielded sophisticated synthetic devices, including particles that serve a particular function (e.g., light control for nanoactuation [11], performing clocked, multistage logic [12], actuation using external magnetic fields [13], [14]), but not particles possessing autonomous circuitry, logic manipulation, and information storage [1]. Besides the work published in [1], [9], existing micro- or nanoparticles do not autonomously process information when decoupled from their environment [15], [16]. The particles created in Koman and Liu et al. [1], [9] are the basis for the synthetic cells proposed in this paper, and we will make the following assumptions based on this work:

- 1) Synthetic cells can guide their own motion either mechanically [17], by means of elaborate swimming strategies like rotating helical flagella [18], or (more likely) chemically [19], [20], through use of Pt-Au bimetallic rods [21], self-electrophoresis [22], [23], self-diffusiophoresis [24], or self-thermophoresis [25].
- 2) Synthetic cells can send and receive communications optically, using integrated LEDs [26] and solid-state or organic light emitting diodes [26], [27].
- 3) Synthetic cells can detect a target by using a chemiresistor to recognize the target’s specific chemical analyte [1], [6], [7].

Lastly, and most importantly, [3] discusses that it is mandatory to guarantee the safety and robustness of biomedical microrobots while they are operating inside a human body. If devices are to be employed in medical applications, they must not damage tissues or cause any negative reaction from the body. One of the primary contributions of the work in this paper is constraining synthetic cells to behave as a physical implementation of a Bayesian update, creating a basis for formal properties and guarantees on their behavior. This ensures robustness in their performance, which can

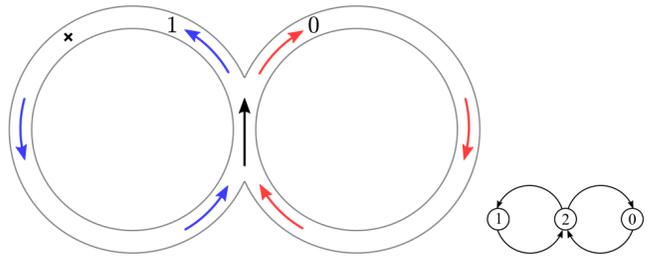


Fig. 2: Left: An example cyclic maze with only two possible paths. Based on the control policy of each two-bit synthetic cell (a 0 or a 1), the cell will follow either the right path or the left path in search of the target  $\times$ . The cell uses its second bit to encode whether or not it has detected the target. Right: A graphical representation of this synthetic cell environment.

be translated to safety guarantees in specific situations and environments.

Bayesian approaches have been applied to reinforcement learning [28], [29], but often in supervised scenarios where there are experts and learners. These methods also often employ passive observations by the learners, rather than explicit communication from experts to learners. In this paper, we employ reinforcement learning of policy updates in rollouts of executions. We also enforce communication between agents, specifically from successful agents to unsuccessful ones when they encounter each other.

## III. PROBLEM DEFINITION

*Environment:* Our goal is to mimic the immune response, so we simulate a model of the circulatory system [30] in Section IV. But in this section, we present a simplified, introductory model. This model consists of a maze, shown in Figure 2, with only two possible paths: left or right. The cells will search for a target, located at the black  $\times$ .

*Policy Execution:* For this introductory example, each synthetic cell has only two bits: one for its control policy (1 for left or 0 for right) and one to indicate whether it has found the target (1) or not (0). Each cell begins with a randomly assigned control policy and loops through the maze. They have a probability of a false positive  $p_{fp}$  (detecting the target when it is *not* there) and a probability of a false negative  $p_{fn}$  (*not* detecting the target when it is there). If a synthetic cell thinks it has detected the target, it changes its second bit, which we will call its *success bit*, to a 1. This policy execution is illustrated in Figure 3.

*Communication:* Using methods of optical information transmission discussed in Section II, synthetic cells are capable of local communication when they are within a certain distance of each other.

When synthetic cells reconvene in the middle of the maze, there is an opportunity for communication. We use a parameter  $\rho$  to characterize how many other synthetic cells, on average, each cell will interact with during one loop of the maze (no matter what control policy or success bit either cell has). This parameter  $\rho$  is related to the density of synthetic cells in the environment, and how likely they are to pass within communication range of each other.

If two synthetic cells come into contact, a successful cell (with a 1 for its success bit) will tell an unsuccessful cell

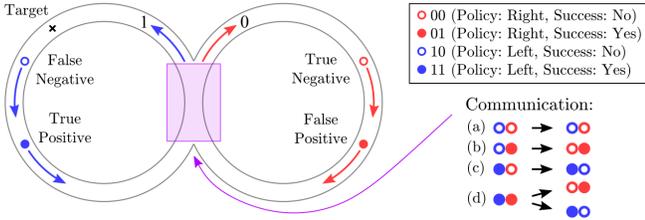


Fig. 3: Each synthetic cell begins with an initial control policy of either a 1 or a 0, which causes it to turn either left or right in the maze. If the cell thinks it has found the target, it changes its second bit (its success bit) to a 1. Due to stochasticity in the environment and the cells, there is the possibility of a false positive or a false negative. Cells might communicate in the middle region, shown in purple. (a) If both cells are unsuccessful they will not communicate any information even if they are within range of each other. (b) and (c) If one cell is successful and one isn't, the successful cell will communicate its policy to the unsuccessful one. (d) If both cells are successful, one (selected randomly) will listen to the other.

(with a 0 for its success bit) its “correct” policy—even if it is successful because of a false positive. If both communicating synthetic cells are successful, one will listen to the other, but which one is the listener is randomly chosen. And if both are unsuccessful they will not tell each other anything. In this way, the success bit also functions as a read/write bit. If it is a 0, the cell will listen to others (read) and if it is a 1, the cell will try to broadcast its policy to others (write). These different scenarios are depicted in Figure 3, and a visualization of this system is shown in the supplementary video.

When synthetic cells enact their simple algorithm of policy execution, possible target detection, and communication, the cells all end up with the policy that passes the target. In Figure 4, the number of synthetic cells with each state are shown as they loop through the maze multiple times and communicate with each other between loops. By the ninth iterate, every synthetic cell has the policy that takes it past the target. This optimal final result always occurs in the case of this simple maze, as long as  $p_{fp}$  and  $p_{fn}$  are sufficiently small and  $\rho > 0$ . But with more complicated environments and possibilities (for example, the maze in Figure 1) it becomes more difficult to ensure this result. To address this, we increase the number of bits on each cell, to extend their capabilities.

#### IV. SIMULATIONS

We now introduce a more complex example, where each synthetic cell has three bits and the maze has four possible paths. The environment is shown in Figure 1, where the possible control policies are: 01, which takes the synthetic cells past the target; 00 and 11, which both loop the cells around the maze; and 10, which leads the cells down a path that they have probability  $p_{lost}$  of never returning from. The goal is for as many cells as possible to end with the 01 policy, where they will all be heading toward the target.

The possibility of getting lost adds further complexity to the system, because not only is the target not reachable with policy 10, but some cells with that policy will not return at all. This could be equivalent to different environmental factors in a body, for example an area with enough acidity

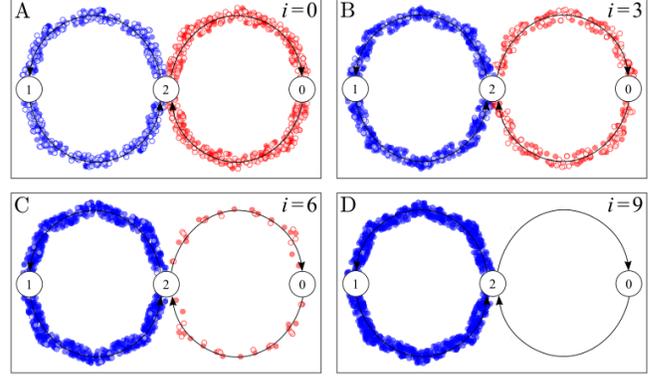
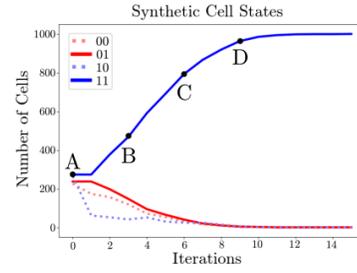


Fig. 4: Top: Results of a simulation with 1000 two-bit synthetic cells for 15 iterations (15 loops around the maze shown in Figure 2), with parameters  $p_{fp} = 0.2$ ,  $p_{fn} = 0.2$ , and  $\rho = 1.0$ . Bottom: Density plots illustrate the distribution of cells at different time increments. As the cells loop through the maze, they converge to the policy that takes them past the target.

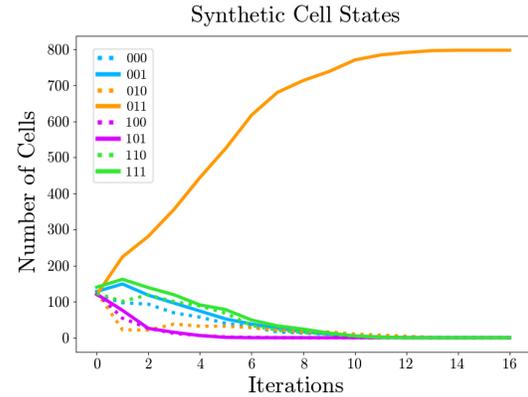


Fig. 5: Simulated results for 1000 three-bit synthetic cells executing their policies and communicating in the maze from Figure 1, with parameters  $p_{fp} = 0.2$ ,  $p_{fn} = 0.2$ ,  $p_{lost} = 0.5$ , and  $\rho = 1$ . Around 800 of them converge to the correct policy where they will find the target.

to damage or destroy synthetic cells. In practice, we predict that there will be many opportunities for synthetic cells to veer off course and get lost, or to get stuck such that they can no longer contribute to the goals of the group. As the magnitude of  $p_{lost}$  increases, more cells get lost and fewer are able to reach the target and combat an invading antigen.

Figure 5 shows results for 1000 three-bit synthetic cells exploring the environment shown in Figure 1. All but the cells that have been lost converge to the correct policy after 12 iterations.

#### Particle Filter

A particle filter is a nonparametric implementation of a Bayes filter that represents a distribution using a set of

---

## Algorithm Comparison

---

### Particle Filter

**Prior** distribution is described by  $L$  particles and their weights  
 $X_n, w_n$

Distribution is sampled, resulting in  $L$  **new particles**

$$\bar{x}_{n+1}^1, \bar{x}_{n+1}^2, \dots, \bar{x}_{n+1}^L$$

Based on a **measurement**, weights<sup>1</sup> are assigned to each particle

$$w_{n+1}^\ell = P(z_{n+1} | \bar{x}_{n+1}^\ell)$$

**Resample** by drawing with replacement  $L$  particles from weighted set  $\bar{X}_{n+1}$

**Posterior** distribution is described by the resampled particles and their weights

$$X_{n+1}, w_{n+1}$$

### Synthetic Cell Implementation

**Prior** distribution is described by  $M$  cell policies and success bits

$$Y_n, s_n$$

Cells execute their policies, resulting in  $M$  **new states**

$$\bar{y}_{n+1}^1, \bar{y}_{n+1}^2, \dots, \bar{y}_{n+1}^M$$

Based on its **success bit**, a cell might broadcast its policy

$$s_{n+1}^m = 0 \text{ or } 1$$

Each cell  $\bar{y}_{n+1}^m$  **communicates** with  $\rho$  other cell(s). If any cell has  $s_{n+1}^m = 1$ , some cell(s) will change their policy.

This approximates resampling as  $\rho \rightarrow M$ .

**Posterior** distribution is described by the final synthetic cell policies and success bits

$$Y_{n+1}, s_{n+1}$$

---

<sup>1</sup>In the case of this synthetic cell example, the measurement  $z_{n+1}$  is the number of cells with each policy. The weight (likelihood of measurement  $z_{n+1}$  occurring if hypothesis  $\bar{x}_{n+1}^\ell$  is correct) is calculated by the number of observed particles with the same policy as  $\bar{x}_{n+1}^\ell$  divided by  $L$ .

---

random samples drawn from that distribution [31], [32]. In a particle filter, the samples from the distribution are called *particles*. We denote these samples  $X_n := x_n^1, x_n^2, \dots, x_n^L$ . Each particle  $x_n^\ell$  is a hypothesis of the true world state at time  $n$ —in our example, each particle would be a hypothesis of the policy that leads to the target.

The most basic variant of a particle filter algorithm begins with the particle set  $X_n$  and weights  $w_n$ , which together represent a prior distribution. This distribution is sampled, resulting in  $L$  particles  $\bar{x}_{n+1}^1, \dots, \bar{x}_{n+1}^L$ . The bar indicates that these samples are taken before the measurement has been incorporated. Next, a measurement  $z_{n+1}$  is obtained, and it is used to calculate new weights  $w_{n+1}^\ell$  for each particle. The weight is the probability of the measurement given each particle,  $w_{n+1}^\ell = P(z_{n+1} | \bar{x}_{n+1}^\ell)$ . Lastly, the particle filter resamples the distribution by drawing with replacement  $L$  particles from the weighted set  $\bar{X}_{n+1}$ , where the probability of drawing each particle is given by its weight. The resampled particles  $X_{n+1} = x_{n+1}^1, \dots, x_{n+1}^L$ , along with the weights  $w_{n+1}$ , represent the posterior distribution—an updated estimate of which policy leads to the target. Note that in the case of the synthetic cell ensemble, the particle filter is estimating discrete states: each particle can only take one of four different values. There are much more than four particles, so many particles will hypothesize that the target is at the same state.

For the synthetic cell implementation, we begin with random policies (and random success bits) on all of the synthetic cells, similar to starting with a uniformly distributed prior distribution. This distribution of  $M$  synthetic cells has discrete states  $Y_n := y_n^1, y_n^2, \dots, y_n^M$ . The cells execute their policies and some return with a success bit, resulting in new cell states  $\bar{y}_{n+1}^1, \dots, \bar{y}_{n+1}^M$ . The value of the success bit  $s_{n+1}^m$  of each cell  $\bar{y}_{n+1}^m$  is a binary implementation of the weight  $w_{n+1}^m$  in a particle filter. Instead of calculating a conditional probability so that the weights are *between* 0 and 1, the weights *are* either 0 or 1 (before being normalized by

the total number of successful cells). A cell  $\bar{y}_{n+1}^m$  with a 0 success bit will not communicate its policy to any other cells, meaning, in particle filter terms, that it will not be sampled from—so its weight is effectively  $w_{n+1}^m = 0$ .

For example, consider a situation where there are  $M$  synthetic cells (and  $M_{001}$  synthetic cells with policy 00 and a 1 for a success bit,  $M_{110}$  cells with policy 11 and a 0 for a success bit, etc.) and  $\rho = 1$ , meaning that each cell will communicate with one other cell during each loop around the maze. Cell  $\bar{y}_{n+1}^m$  has a  $\frac{\rho}{M}$  probability of communicating with any other cell  $\bar{y}_{n+1}^i$ , where  $1 \leq i \leq M$ , during a given cycle. Cell  $\bar{y}_{n+1}^m$ 's probability of sampling a cell with policy 00 is  $\frac{M_{001}}{M}$ , its chance of sampling a cell with policy 01 is  $\frac{M_{011}}{M}$ , and so on. It also has a chance of staying the same, anytime it communicates with a cell with a 0 success bit, which occurs with probability  $\frac{M_{000} + M_{010} + M_{100} + M_{110}}{M}$ .

This is the main difference between the particle filter algorithm and the synthetic cell implementation: the synthetic cells have some probability of *not* resampling, and just staying the same—unlike particles in a particle filter which are all resampled, every iteration. This difference is demonstrated in the Algorithm Comparison box, above. If each cell *always* communicated with a random successful cell, its behavior would be the same as that of a particle filter. This is illustrated in Figure 6. The far right panel of Figure 6 shows a particle filter applied to the synthetic cell system. There are  $L = 1000$  particles being randomly sampled from the synthetic cell distribution (which is also comprised of  $M = 1000$  cells), and the weights are being updated based on observations of synthetic cell policies. As  $\rho$  increases, the amount of resampling increases, and the synthetic cell behavior is guaranteed to converge to the particle filter behavior. The physical execution of the group of synthetic cells approximates the particle filter algorithm.

Similarly, a particle filter approximates a Bayes filter. The approximation error of a particle filter approaches zero as the number of particles goes to infinity—the error depends

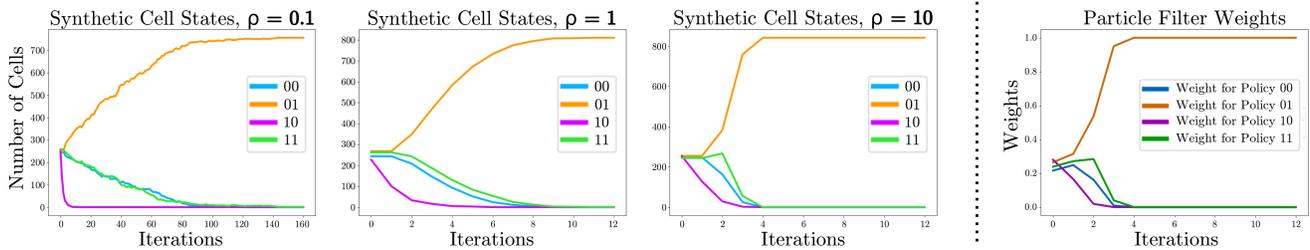


Fig. 6: Synthetic cell executions for different values of  $\rho$  for 1000 synthetic cells and  $p_{fp} = 0.1$ ,  $p_{fn} = 0.1$ ,  $p_{lost} = 0.5$ . The rightmost panel shows a particle filter implementation for this system, where the measurements are the current states of the synthetic cells. As  $\rho$  increases, it approximates the particle filter. Note that the third plot, where  $\rho = 10$ , is nearly identical to the plot of the particle filter weights.

on the number of particles, not on the resampling. In fact, an alternative version of a particle filter does not resample at all [32]. Since the asymptotic guarantee on a particle filter approximating a Bayes filter does not depend on resampling, it consequently holds for synthetic cells as well. Therefore, since the synthetic cell implementation approximates a particle filter, and a particle filter approximates a Bayesian update, we can conclude that a synthetic cell system using this algorithm approximates a Bayesian update.

This result, which is illustrated in Figure 6, guarantees convergence properties for how synthetic cells will probabilistically behave. These guarantees are valuable because they can be used to reliably predict how synthetic cells will perform in new scenarios, and we can be certain of robustness and safety requirements for physical experiments.

### Model of the Human Circulatory System

Many models of the human cardiovascular system exist, including a 36 vessel body tree [33], a lumped parameter model [34], and a mathematical model featuring both linear and nonlinear constitutive relations [35]. In this paper, we use the model from Hardy et al. [30], which clearly defines the 24 different chambers in the circulatory system, as well as the connections going into and out of each one. This model is shown in Figure 7, where each number represents a chamber, as described in the legend, and the connections depict inputs and outputs for blood flow. Figure 8 shows how the graphical representation of the circulatory system can be illustrated as the same type of maze that was shown in Figures 1 and 2.

To navigate in this environment, synthetic cells require seven bits. This comes from the seven way intersection at node 2, where cells need at least three bits to choose a path; the four way intersection at node 12, where they each need two more bits to decide on a path; and two more bits to use as success bits (the reason *two* success bits are required will be discussed in the next section). The policy bit organization is shown in Figure 8.

We simulated synthetic cell executions in this scenario, where the desired target was in the leg, specifically node 13. In this circulatory system model, there are many different policies that will lead to finding the target. It doesn't matter how the cells pass through the pulmonary system (states 2–9 in Figure 7 or the top part of the maze in Figure 8), as long as they reach the leg in the end. The results of this simulation are shown in Figure 9.

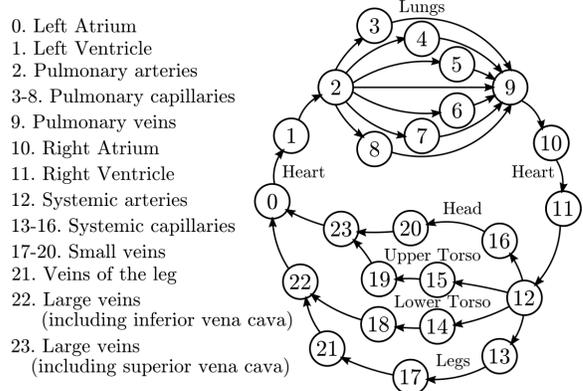


Fig. 7: Adapted from Figure 1, Figure 2, Figure 3, and Table 2 in Hardy et al. [30]. Each number represents a chamber, as described in the legend. The connections between chambers are inputs and outputs illustrating blood flow.

In the previous example, shown in Figures 1 and 5, we simulated a probability of getting lost along one of the paths. This was to acknowledge that in practice, unexpected events can happen where some synthetic cells will get lost, stuck, destroyed, or otherwise do not contribute to the group's estimate of the target location. We recognize that this can happen no matter where the cell is, so in this example we implemented a small  $p_{lost}$  on every execution of every synthetic cell. All of the cells, besides the ones that have been lost to the environment due to  $p_{lost}$ , converge to the correct policy by about the twenty sixth iteration. A visualization of this system is shown in the supplementary video.

### Multiple and Moving Targets

In earlier sections, our algorithm was shown to enable all simulated synthetic cells to converge to a policy that passed by a single target. But there will not always be single, stationary targets in the immune system. In this section, we use the same algorithm to show that multiple and moving targets can be found and communicated, without any prior knowledge of the number or movements of the targets.

If two targets in the same environment are in series<sup>1</sup>, the cell needs to be able to distinguish between them. One

<sup>1</sup>Here, series and parallel mean the same things as they do in circuitry: if two nodes are in series a cell can flow through both of them (e.g., node 3 and node 13, in Figure 7), and if they are in parallel a cell cannot (e.g., node 13 and node 20).

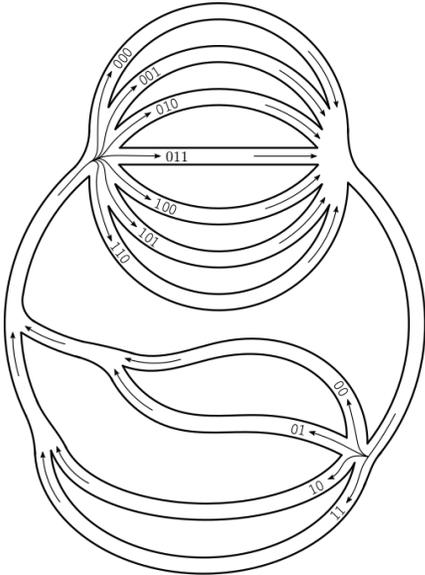


Fig. 8: A maze, similar to those in Figures 1 and 2, based on the inputs and outputs of chambers in the circulatory system, shown in Figure 7. Bit assignments for each path are also shown, to illustrate the 5 bit policies that describe each of the 28 possible paths through the system.

synthetic cell could be passing by a target in node 3, while another cell is passing by a target in node 13. If they both think they have the correct policy, they might never learn to pass by both targets. If they have separate success bits for each intersection, they ensure that they are reaching as many targets as possible. This conclusion can be expressed in the following equation for the number of bits,  $B$ , required for any cyclic graph.

$$B = \sum_{i=1}^I \text{ceil}(\log_2(P_i)) + I \quad (1)$$

In Eq. 1,  $B$  is the number of bits required to navigate the graph,  $I$  is the number of intersections, or diverging nodes (nodes that have multiple edges leaving them), and  $P_i$  is the number of edges leaving each intersection. The ceiling function  $\text{ceil}$  rounds up to the nearest integer, as we only consider entire bits.

The circulatory system shown in Figure 7 has  $I = 2$  intersections, at nodes 2 and 12, which have 7 and 4 outgoing edges, respectively, and therefore  $B = 7$  bits are required to solve the graph.

Figure 10 shows simulated results for 1000 synthetic cells navigating through an environment that has multiple targets: at nodes 3, 13, and 20. Within twenty cycles, the cells learn policies to pass by as many targets as possible (in this environment, each cell can pass by a maximum of two targets as they have two decision points throughout the graph).

Next, we investigate moving targets. Once all of the synthetic cells have converged upon a target's location, what if it moves somewhere else? To find it again, the cells will have to rely on a small amount of random exploration and decaying memory. We model the random exploration as a very small chance of one of a cell's bits flipping at any moment (this could also be considered a cell making a

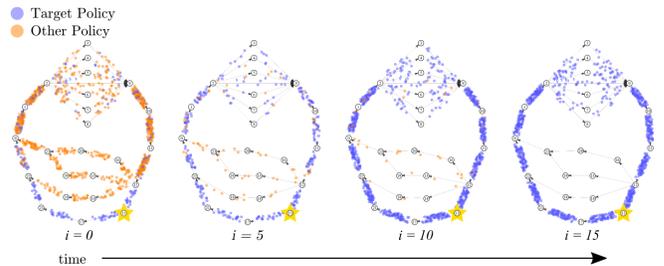


Fig. 9: Density plots illustrating the distribution of 1000 seven-bit synthetic cells executing their policies in the maze from Figure 8, with parameters  $p_{fp} = 0.1$ ,  $p_{fn} = 0.1$ , and  $\rho = 1.0$ . The target is shown by a yellow star and the cells with policies that pass by it are shown in blue.

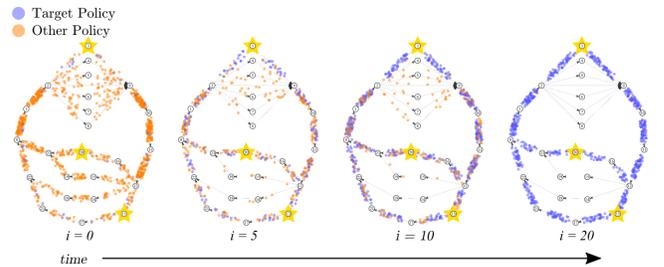


Fig. 10: Density plots illustrating the distribution of 1000 seven-bit synthetic cells executing their policies in the maze from Figure 8, with parameters  $p_{fp} = 0.1$ ,  $p_{fn} = 0.1$ , and  $\rho = 1.0$ . The targets are at nodes 3, 13, and 20, shown by yellow stars.

mistake). Decaying memory enables synthetic cells' success bits to turn off (back to 0) after some amount of time has passed since they last detected a target. We know from [1], [9] that this is physically feasible, given variable chemical decay rates and reactions that act similarly to capacitors with a decaying charge.

Figure 11 shows simulated results for 1000 synthetic cells navigating through an environment and learning the policies to keep finding the new location of a target which moves from node 13 to node 3, and finally to node 20.

## V. CONCLUSIONS

This work demonstrated a novel algorithm for reinforcement learning behavior, in the form of policy updates based on observations, in synthetic cell ensembles. Each synthetic cell only has a few bits of memory and very simple communication abilities. Despite this, we show that the cells can use local algorithms to refine their global belief of how to reach a target location—reflected in the distribution of the control policies of each cell.

This was applied to a model of the human cardiovascular system, where the group of cells was able to converge to the correct policy (apart from those lost to simulated environmental factors), using only seven bits. These same particles were also able to detect and navigate toward multiple targets as well as find and follow a moving target. The result that only seven bits are necessary to function in this model of the circulatory system demonstrates that even with very limited computation synthetic cells are a capable system in an environment as complex as the human body.

We showed that the synthetic cell implementation approximates a particle filter, and that the only difference

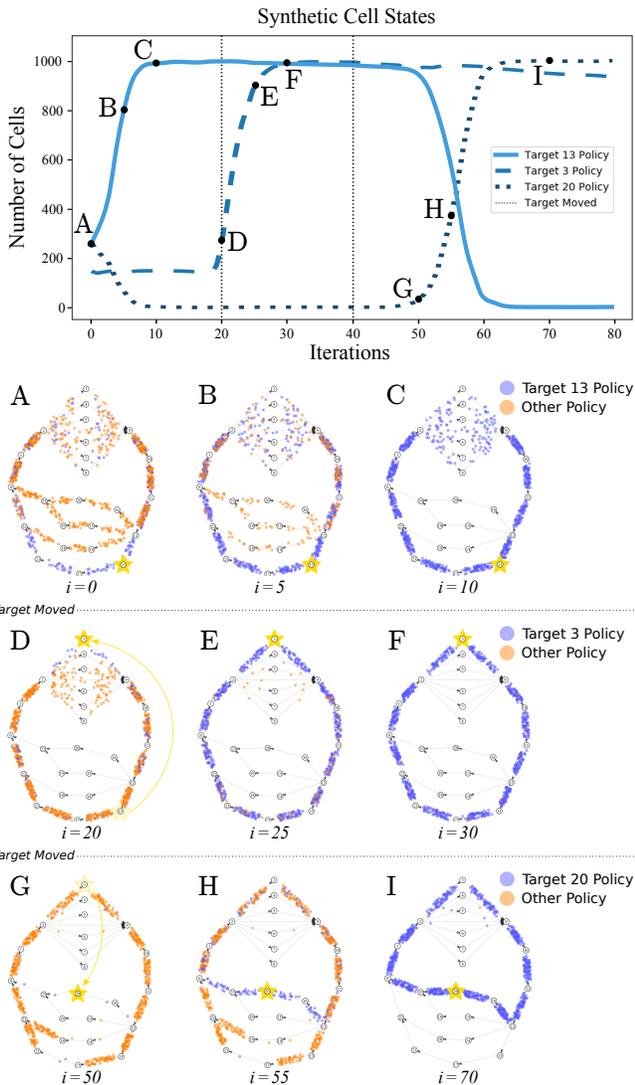


Fig. 11: Simulated results for 1000 seven-bit synthetic cells executing their policies in the maze from Figure 8, with parameters  $p_{fp} = 0.1$ ,  $p_{fn} = 0.1$ , and  $\rho = 1.0$ , as they search for a target that moves from node 13 to node 3 to node 20. At nine instances in time (indicated by letters A-I), a density plot is shown to illustrate the distribution of the cells.

between the two methods is the execution of the resampling step. Since the asymptotic guarantee on a particle filter approximating a Bayes filter depends on the number of particles, and not the resampling, we concluded that the synthetic cell system is a suboptimal Bayesian filter. This result constitutes what might be the first decision theoretic model of the immune system, and provides formal properties for the behavior of this type of synthetic cell ensemble that can be applied in future work with different tasks, environments, and decision variables.

In the future, we will pursue different elements of this work, including encoding the need to explore (e.g., if the correct policy is not known to any cells in the ensemble, or if the target has been successfully destroyed). To do this we will borrow methods from [36], which demonstrates effective algorithms for path planning of long excursions that agents may not return from. We also intend to implement these

results experimentally in the near future.

#### ACKNOWLEDGMENTS

We greatly appreciate Albert Tianxiang Liu and Michael Strano for their feedback on this manuscript.

#### REFERENCES

- [1] V. B. Koman, P. Liu, D. Kozawa, A. T. Liu, A. L. Cottrill, Y. Son, J. A. Lebron, and M. S. Strano, "Colloidal nanoelectronic state machines based on 2D materials for aerosolizable electronics," *Nature Nanotechnology*, vol. 13, pp. 819–827, Sep. 2018.
- [2] D. Seo, R. M. Neely, K. Shen, U. Singhal, E. Alon, J. M. Rabaey, J. M. Carmena, and M. M. Maharbiz, "Wireless Recording in the Peripheral Nervous System with Ultrasonic Neural Dust," *Neuron*, vol. 91, no. 3, pp. 529–539, 2016.
- [3] M. Sitti, H. Ceylan, W. Hu, J. Giltinan, M. Turan, S. Yim, and E. Diller, "Biomedical Applications of Untethered Mobile Milli/Microrobots," *Proceedings of the IEEE*, vol. 103, no. 2, pp. 205–224, Feb. 2015.
- [4] D. D. Chaplin, "Overview of the immune response," *The Journal of Allergy and Clinical Immunology*, vol. 125, pp. S3–23, Feb. 2010.
- [5] T. Newman, "How the immune system works," *Medical News Today*, Jan. 2018.
- [6] P. Liu, A. T. Liu, D. Kozawa, J. Dong, J. F. Yang, V. B. Koman, M. Saccone, S. Wang, Y. Son, M. H. Wong, and M. S. Strano, "Autoperforation of 2D materials for generating two-terminal memristive Janus particles," *Nature Materials*, vol. 17, pp. 1005–1012, Oct. 2018.
- [7] P. Liu, A. L. Cottrill, D. Kozawa, V. B. Koman, D. Parviz, A. T. Liu, J. Yang, T. Q. Tran, M. H. Wong, S. Wang, and M. S. Strano, "Emerging trends in 2D nanotechnology that are redefining our understanding of "Nanocomposites"," *Nano Today*, 2018.
- [8] A. Pervan and T. D. Murphey, "Low complexity control policy synthesis for embodied computation in synthetic cells," *The 13th International Workshop on the Algorithmic Foundations of Robotics*, Dec. 2018.
- [9] A. T. Liu, J. F. Yang, L. N. LeMar, G. Zhang, A. Pervan, T. D. Murphey, and M. S. Strano, "Autoperforation of two-dimensional materials to generate colloidal state machines capable of locomotion," *Submitted*.
- [10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998.
- [11] T. Ding, V. K. Valev, A. R. Salmon, C. J. Forman, S. K. Smoukov, O. A. Scherman, D. Frenkel, and J. J. Baumberg, "Light-induced actuating nanotransducers," *Proceedings of the National Academy of Sciences*, vol. 113, no. 20, pp. 5503–5507, 2016.
- [12] J. Yao, H. Yan, S. Das, J. F. Klemic, J. C. Ellenbogen, and C. M. Lieber, "Nanowire nanocomputer as a finite-state machine," *Proceedings of the National Academy of Sciences*, vol. 111, no. 7, pp. 2431–2435, 2014.
- [13] M. Sitti, "Voyage of the microrobots," *Nature*, vol. 458, pp. 1121–1122, Apr. 2009.
- [14] L. Zhang, J. J. Abbott, L. Dong, B. E. Kratochvil, D. Bell, and B. J. Nelson, "Artificial bacterial flagella: Fabrication and magnetic control," *Applied Physics Letters*, vol. 94, no. 6, p. 064107, 2009.
- [15] C. Kaewsaneha, P. Tangboriboonrat, D. Polpanich, and A. Elaissari, "Multifunctional Fluorescent-Magnetic Polymeric Colloidal Particles: Preparations and Bioanalytical Applications," *ACS Applied Materials & Interfaces*, vol. 7, no. 42, pp. 23 373–23 386, 2015.
- [16] N. Kamaly, B. Yameen, J. Wu, and O. C. Farokhzad, "Degradable Controlled-Release Polymers and Polymeric Nanoparticles: Mechanisms of Controlling Drug Release," *Chemical Reviews*, vol. 116, no. 4, pp. 2602–2663, 2016.
- [17] M. Guix, C. C. Mayorga-Martinez, and A. Merkoj, "Nano/Micromotors in (Bio)chemical Science Applications," *Chemical Reviews*, vol. 114, no. 12, pp. 6285–6322, 2014.
- [18] S. C. Lenaghan, S. Nwandu-Vincent, B. E. Reese, and M. Zhang, "Unlocking the secrets of multi-flagellated propulsion: drawing insights from *Trichomonas foetus*," *Journal of the Royal Society, Interface*, vol. 11, Apr. 2014.
- [19] J. L. Moran and J. D. Posner, "Locomotion of Electrocatalytic Nanomotors due to Reaction Induced Charge Auto-Electrophoresis," *Physical Review E*, vol. 81, Jun. 2010.
- [20] J. L. Moran, *Robotic Systems and Autonomous Platforms*. Woodhead Publishing, 2019.

- [21] W. F. Paxton, K. C. Kistler, C. C. Olmeda, A. Sen, S. K. St. Angelo, Y. Cao, T. E. Mallouk, P. Lammert, and V. H. Crespi, "Catalytic nanomotors: autonomous movement of striped nanorods," *Journal of the American Chemical Society*, vol. 126, p. 1342413431, 2004.
- [22] Y. He, J. Wu, and Y. Zhao, "Designing Catalytic Nanomotors by Dynamic Shadowing Growth," *Nano Letters*, vol. 7, no. 5, pp. 1369–1375, 2007.
- [23] J. L. Moran, P. M. Wheat, and J. D. Posner, "Phoretic Self-Propulsion," *Annual Review of Fluid Mechanics*, vol. 49, pp. 511–540, 2017.
- [24] J. R. Howse, R. A. L. Jones, A. J. Ryan, T. Gough, R. Vafabakhsh, and R. Golestanian, "Self-Motile Colloidal Particles: From Directed Propulsion to Random Walk," *Physical Review Letters*, vol. 99, p. 048102, Jul. 2007.
- [25] H.-R. Jiang, N. Yoshinaga, and M. Sano, "Active Motion of a Janus Particle by Self-Thermophoresis in a Defocused Laser Beam," *Physical Review Letters*, vol. 105, Dec 2010.
- [26] X. Wu, I. Lee, Q. Dong, K. Yang, D. Kim, J. Wang, Y. Peng, Y. Zhang, M. Saliganc, M. Yasuda, K. Kumeno, F. Ohno, S. Miyoshi, M. Kawaminami, D. Sylvester, and D. Blaauw, "A 0.04MM316NW Wireless and Batteryless Sensor System with Integrated Cortex-M0+ Processor and Optical Communication for Cellular Temperature Measurement," in *2018 IEEE Symposium on VLSI Circuits*, June 2018, pp. 191–192.
- [27] S. Lee, A. J. Cortese, P. Trexel, E. R. Agger, P. L. McEuen, and A. C. Molnar, "A  $330\mu\text{m} \times 90\mu\text{m}$  opto-electronically integrated wireless system-on-chip for recording of neural activities," in *2018 IEEE International Solid-State Circuits Conference, ISSCC*, 2018, pp. 292–294.
- [28] B. Price and C. Boutilier, "A Bayesian Approach to Imitation in Reinforcement Learning," *International Joint Conferences on Artificial Intelligence*, 2003.
- [29] D. Ramachandran and E. Amir, "Bayesian Inverse Reinforcement Learning," in *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, 2007, pp. 2586–2591.
- [30] H. H. Hardy, R. E. Collins, and R. E. Calvert, "A digital computer model of the human circulatory system," *Medical and Biological Engineering and Computing*, vol. 20, no. 5, pp. 550–564, Sep. 1982.
- [31] C. M. Bishop, *Pattern Recognition and Machine Learning*. Berlin, Heidelberg: Springer-Verlag, 2006.
- [32] D. F. Sebastian Thrun, Wolfram Burgard, *Probabilistic Robotics*. The MIT Press, 2005.
- [33] M. Mirzaee, O. Ghasemalizadeh, and B. Firoozabadi, "Modeling Vessels in Human Cardiovascular System and Calculating Pressure Wastes Using Lumped Method," *The Internet Journal of Bioengineering*, vol. 3, 2008.
- [34] Y.-T. Kim, J. S. Lee, C.-H. Youn, J.-S. Choi, and E. B. Shim, "An integrative model of the cardiovascular system coupling heart cellular mechanics with arterial network hemodynamics," *Journal of Korean Medical Science*, vol. 28, no. 8, pp. 1161–1168, Aug. 2013.
- [35] M. J. Conlon, D. Russell, and T. Mussivand, "Development of a Mathematical Model of the Human Circulatory System," *Annals of Biomedical Engineering*, vol. 34, pp. 1400–13, 10 2006.
- [36] M. Otte and D. Sofge, "Path planning for information gathering with lethal hazards and no communication," *The 13th International Workshop on the Algorithmic Foundations of Robotics*, Dec. 2018.