2020 American Control Conference
Denver, CO, USA, July 1-3, 2020

Reinforcement Learning for Elimination of Reentrant Spiral Waves in Excitable Media

James K. Senter[1†], D. Wilson[1*], and Amir Sadovnik[1‡]

Abstract— **Despite recent advancements in understanding the mechanisms underlying sudden cardiac death due to cardiac fibrillation, new defibrillation techniques have been slow to manifest. The reasons for this are manifold, but from a controls perspective, the spatiotemporal behavior exhibited by the electrical activity of the heart during fibrillation is high-dimensional, chaotic, and fundamentally nonlinear making standard control techniques difficult to implement. In this work, we investigate the use of a reinforcement learning framework to identify a control strategy to eliminate reentrant spiral waves that are associated with cardiac fibrillation. We propose a reduced order model that replicates the behavior of an idealized spiral wave core traveling in an excitable medium. We implement the Q-learning method with function approximation using a neural network to learn a control strategy that actively drives a spiral core to the boundary of the domain where it can be absorbed. Results indicate that the reinforcement learning algorithm is able to rapidly learn an effective control strategy for use in the reduced order model. Continued development of this framework for implementation in more realistic models could inform the design of active control strategies to achieve low-energy control of spatiotemporal chaos in the heart associated with cardiac arrest.**

I. Introduction

Cardiac arrest, caused by fibrillation occurring in the lower chambers of the heart, is a particularly deadly manifestation of heart disease occurring in more than 300,000 patients annually in the United States [1]. It is well established that cardiac arrest is caused by electrical turbulence resulting from self-sustaining spiral waves in the heart [2], [3]. High energy shocks remain the most clinically effective way to eliminate fibrillation, however, these shocks cause severe secondary side effects such as electroporation, intense pain, and additional damage to already diseased hearts [4].

More recently, alternative low-energy and pain-free approaches to defibrillate have been proposed. For instance, antitachycardia pacing [5] represents a promising strategy whereby traveling waves emanating from a point source can overtake and replace the reentrant spiral waves. Additionally, the application of multiple low intensity defibrillating pulses instead of one large shock has been shown to reduce the energy threshold required for successful defibrillation [6], [7], [8].

While much progress has been made on these aforementioned open-loop defibrillation strategies, effective feedback control strategies for elimination of spiral waves have been slower to manifest. One significant barrier to the development of feedback control strategies is the dynamical complexity of fibrillation: spiral waves dynamics during fibrillation are often dynamically unstable, being created and disappearing in an irregular manner [9], [10]. Additionally, the dynamical behavior of spirals is significantly influenced by the geometry of an individual's heart [11], [12] which would necessitate a personalized and adaptable control strategy in a clinical setting.

In light of these considerations, in this paper we investigate the feasibility of using a reinforcement learning framework to actively learn and implement a strategy for controlling the behavior of reentrant spiral waves. The organization of this paper is as follows: Section II gives background information on the behavior of spiral waves in excitable media and proposes a reduced order model that represents the behavior of an idealized spiral wave core. Section III describes the reinforcement learning algorithm that will be used to learn an effective control strategy to guide spiral wave cores to an inexcitable tissue boundary (through which it can be absorbed and eliminated). Section IV shows results of the learning algorithm and describes the learned control strategy. While results are promising, additional work must be done in order make the control problem more applicable to the problem of eliminating cardiac arrest in real cardiac tissue and Section V gives concluding remarks highlighting opportunities for extension in future work.

II. Prototype Problem: Spatiotemporal Control of a Idealized Spiral Core

Numerical models of cardiac tissue vary greatly in terms of their physiological complexity. A commonality among most models is that they characterize the spatiotemporal evolution of transmembrane voltage subject to the local dynamics of ion concentrations and gating variables. In many cases, it is possible to replicate complicated electrical behaviors observed in cardiac tissue using a so-called monodomain model [2]

$$C_m \frac{\partial}{\partial t} u(x,y,t) = D\Delta u(x,y,t) - I_{\text{ion}}(x,y,t) + \sum_{j=1}^{N} I_j(x,y,t). \quad (1)$$

Above, x and y are coordinates on a 2D spatial domain, D is a matrix of diffusion coefficients, Δ is the Laplacian, u corresponds to the transmembrane voltage, C_m is the membrane capacitance, I_{ion} is a membrane current density that depends on the local cellular dynamics, and I_j is a

[1]Department of Electrical Engineering and Computer Science, University of Tennessee, Knoxville, TN 37996, USA
† jsenter3@utk.edu
* dwilso81@utk.edu
‡ asadovnik@utk.edu

current controlled input applied by an external stimulator with N being the total number of stimulators. Eq. (1) supports self sustaining spiral wave patterns that are associated with the emergence of cardiac fibrillation. The primary goal of this work is to identify an effective control strategy to eliminate a spiral wave using a small number of electrical stimualtors.

To approach this problem using a reinforcement learning framework, we must decide how to define our state space on which to take actions. A naive way of doing this would be to take the raw voltages of the membrane as our state. However, this approach would use a very high-dimensional representation of the state. The agent would need to learn how to interpret all of the available data and translate it into useful information from which it can decide on the correct action. Similar to other problems in computer vision, learning will be prohibitively slow under these conditions. Therefore, using model reduction techniques to simplify the problem and operate in a lower dimensional state space is critical for this application.

In order to simplify the problem, we will consider a reduced order model inspired by other reduction frameworks [13], [14], [15], [16], [17] which seek to understand the behavior of Eq. (1) using center manifold theory. Such analysis generally reduces the behavior of a spiral wave to a rigidly rotating core subject to spatial translation (see Figure 1A for example). As a prototype model, we will assume that the behavior of a single spiral wave can be modeled according to:

$$\begin{aligned}
\dot{x}_c &= \alpha\sin(\theta_c) + \sum_{j=1}^{N} \rho_j(t)\langle I_j(x,y), S_x(x,y,x_c,y_c,\theta_c)\rangle,\\
\dot{y}_c &= \alpha\cos(\theta_c) + \sum_{j=1}^{N} \rho_j(t)\langle I_j(x,y), S_y(x,y,x_c,y_c,\theta_c)\rangle,\\
\dot{\theta}_c &= -1 + \sum_{j=1}^{N} \rho_j(t)\langle I_j(x,y), S_\theta(x,y,x_c,y_c,\theta_c)\rangle. \qquad (2)
\end{aligned}$$

Above, x_c and y_c give the spatial location of a spiral core on a 2D domain, $\theta_c \in [0,2\pi)$ represents the orientation of the spiral core, α sets the nominal translation rate of the core, $I_j(x,y)$ represents the spatial influence of an exogenous stimulator and $\rho_j(t)$ gives its magnitude as a function of time, S_A for $A = \{x,y,\theta\}$ represent sensitivity functions of the state variables to exogenous perturbation, and $\langle\cdot,\cdot\rangle$ denotes the L^2 inner product. In the examples to follow, we take $\alpha = 0.1$, $I_j(x,y) = \exp(-50r_j^2)$ for $j = 1,\ldots,N$ where $r_j^2 = (x-x_j)^2 + (y-y_j)^2$, with x_j and y_j being the location of the j^{th} stimulator. Additionally, $S_x = \exp(-200r_c^2)\sin(\theta_p+\theta_c)$ where $r_c^2 = (x-x_c)^2 + (y-y_c)^2$, $\theta_p = \text{atan2}(y-y_c, x-x_c)$, and atan2 is the signed arctangent function. Likewise, $S_y(x,y) = \exp(-200r_c^2)\cos(\theta_p+\theta_c)$ and $S_\theta(x,y) = -\exp(-200r_c^2)\sin(\theta_p+\theta_c)$. Note that this choice of reduced model parameters and functions is not directly derived from any particular model, but that Eq. (2) can be used to qualitatively replicate the perturbed behavior of models of the form Eq. (1). For instance, the perturbed behavior of the Barkley model (with voltage-like variable u and gating variable v) is shown in Figure 1A. The large-core parameters as described in [18] are used to simulate Eq. (1) on a square domain with side length 13.4 units with D taken to be the identity matrix. Simulations are performed on 400×400 discretized grid. This model supports spiral waves with cores (defined as the intersection of the $u = 0.2$ and $v = 0.2$ level set) that have a circular trajectory when $I_{\text{stim}} = 0$ (black line). When taking $I_{\text{stim}}(x,y) = -0.5$ (resp., $+0.5$) in a circle of radius 2.2 centered in the middle of the square (and zero elsewhere) and lasting 0.5 time units, the resulting core trajectory is shown by the red (resp. blue) traces. Qualitatively similar behaviors can be observed in the prototype model on a square domain with side length 1. Here, placing a single stimulator at $(x_1,y_1) = (0.5,0.5)$ and applying a perturbation of $\rho_1 = -40$ (resp., $+40$) lasting 1 time unit yields the trajectory shown in red (resp. blue). In the results to follow, in an effort to focus on the challenges associated with the reinforcement learning paradigm, Eq. (2) model will be used as a surrogate for Eq. (1).

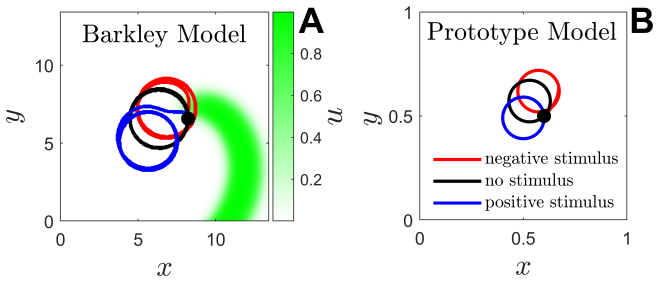

Fig. 1. Comparisons between the monodomain model (panel A) Eq. (1) and the prototype model (panel B) Eq. (2). The perturbed behavior of the prototype model is qualitatively similar to the spiral core trajectory of Eq. (1) when using the Barkley model from [18] to define $I_{\text{ion}}(x,y)$.

III. Reinforcement Learning Model

Using this reduced ordered model, we seek to learn and implement a control strategy for (2) that is able to remove the core from the domain. We therefore model the process as a Markov Decision Process (MDP), and use reinforcement learning to find a suitable strategy.

Assume an agent can observe the current state of the process $s_t \in \mathbb{R}^d$. In our case the state should be able to describe the location and orientation of the spiral core. The agent also has a set of action it can take $\mathscr{A} = \{a_1, a_2, \ldots a_k\}$. The goal of the agent is to take the optimal action for a given state in order to eliminate the core producing a sequence of states and actions: $s_1, a_1, s_2, a_2, \ldots s_T$.

We therefore would like the agent to learn a policy $\pi_*(s_n) = a_n$, where a_n is the optimal action for state s_n. We do this by defining a reward signal r_t, which is sent to the agent after each action it performs. That is, we have a sequence of states actions and rewards: $s_1, a_1, r_1 s_2, a_2, r_2, \ldots s_T$. The goal of the agent is to learn a policy which maximizes the return, which is the discounted sum of all future rewards:

$$\mathscr{R}_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_t, \qquad (3)$$

where T is the final step of our process and γ is a discount factor which describes how much weight we give future rewards. Using this definition of return we can define an action-value function as follows:

$$Q_\pi(s,a) = \mathbb{E}[\mathscr{R}_t | s_t = s, a_t = a]. \quad (4)$$

The above function describes the expected return value when using a certain policy $\pi(s)$. Our learning goal is to find the policy $\pi_*(s)$ which gives us the maximum action value for all states and actions. The optimal action value function is defined as:

$$Q_*(s,a) = \max_\pi Q_\pi(s,a) \quad \forall \quad s \in \mathscr{S}, a \in \mathscr{A}. \quad (5)$$

The optimal action value function follows the Bellman optimality equation:

$$Q_*(s,a) = \mathbb{E}[r_t + \gamma \max_{a'} Q_*(s',a') | s_t = s, a_t = a]. \quad (6)$$

Therefore, one way reinforcement learning tries to learn the optimal policy is by iteratively estimating the optimal action value function. More specifically in this work we use the Q-learning [19] update rule which is an off-policy temporal difference control algorithm. The update rule for Q-learning is:

$$Q(s,a) = Q(s,a) + \alpha \left[r_t + \gamma \max_{a'} Q_*(s',a') - Q(s,a) \right], \quad (7)$$

where α is the step size which defines how much weight will be given to the current update. As shown in [19] as long as all state-action pairs are continually visited, this iterative approach is guaranteed to converge to the optimal action value function $Q_*(s,a)$. However, as in our case the state vector is continuous it is unfeasible to visit each state. Therefore, we try to learn $Q_*(s,a)$ using a function approximator. In this work we use a neural network as our approximator, where our learning goal is to try and minimize the mean square error between the output of the network and our desired output as calculated by Eq. (7). That is, our loss is defined as:

$$L = \mathbb{E}\left[(r_t + \gamma \max_{a'} Q_*(s',a') - Q(s,a))^2 \right]. \quad (8)$$

We can then use methods like stochastic gradient descent to learn the optimal weights. Once we have learned this function we can simply define our optimal policy as $\pi(s) = \max_a Q_*(s,a)$.

When training, the agent must strike a balance between exploration in order to find the best actions, and exploitation in order to take the best actions and reach the goal. We therefore use epsilon-greedy training [19] : with probability ε, the agent takes a random action, and with probability $1-\varepsilon$, the agent takes $\pi(s)$ at state s. ε decays from 1.0 to 0.1 over the course of training. This means that at the beginning of training, the agent explores all actions to find the best starting actions, and then the agent gradually starts following its policy to move closer and closer to the goal while still exploring new routes.

As show in [20], when learning our function approximator it is inefficient to learn from consecutive steps. Therefore we utilize experience replay [21]. Each time the agent takes an action we store the experience $e_t = (s_t, a_t, r_t, s_{t+1})$. This creates a database of experiences. During training we simply select a random batch of experiences and use that in our mini batch gradient descent. This means that experiences will not be consecutive in a single batch and each one can be used more than once during training. As the experience database size is limited we discard old experiences when new ones arrive.

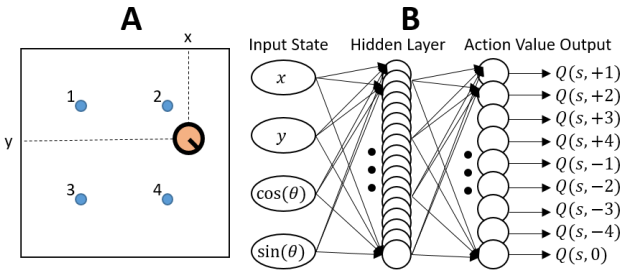

Fig. 2. A diagram of our process model and our reinforcement model. On the left we show our four stimulators (in blue) numbered 1-4 and our core (in orange) with the black line signifying θ. On the right we show our neural network model, with the input state, our hidden layer (128 neurons in our simulations) and our function output $Q(s,a)$. Here $a = +n$ (resp., $a = -n$ corresponds to application of positive (resp., negative) input using stimulator n, and $a = 0$ means all stimulators are off.

In addition, these networks tend to be unstable when updating the parameters of the action value function, if the new weights are directly used for finding the max action value of the next step (the right side in Eq. (7)). Therefore, as is done in [20] we use two separate networks with the same architecture. One network has its weights changed each batch, while the other ones remain frozen. Every $\mathscr{C}$ steps we copy the updated weights to the frozen network and continue in our learning.

A. Implementation

We model the process described in Sec. II using our reinforcement learning notation the following way. Our state $s_t = \{x_t, y_t, \sin(\theta_t)\cos(\theta_t)\}$ is a vector in $\mathbb{R}^4$ which describes the location and orientation of the spiral core. The x_t and y_t inputs are normalized between 0 and 1 so that the model can generalize to a square area of any size. We use the trigonometric functions instead of using θ directly in order to ensure continuity in feature space. That is, the angle $\theta = 2\pi$ should be close both to ε and $2\pi - \varepsilon$ in feature space. Our action set $\mathscr{A}$ is made out of 9 separate possible actions. The first four actions $a_1, ..., a_4$ turn on each of the stimulators with a positive perturbation, the second four actions $a_5, ..., a_8$ do the same with a negative perturbation, and finally the last action a_9 turns all stimulators off. We chose to simplify the space of possible actions by allowing only one stimulator to be activated at a time, for faster training. The 4 stimulators are placed at $(0.25d_x, 0.25d_y)$, $(0.25d_x, 0.75d_y)$, $(0.75d_x, 0.25d_y)$, $(0.75d_x, 0.75d_y)$ where d_x, d_y are the x and y dimensions of our media respectively. See Fig. 2A for a diagram.

It has long been known that self-sustaining spiral waves can be absorbed by an inexcitable tissue boundary and hence eliminated [22], [13]. This mechanism provides a theoretical underpinning for the efficacy of antitachycardia pacing [23]. As such, we will design a reinforcement learning strategy that learns a control policy that drives a core to an inactive tissue boundary where the core can be absorbed and eliminated. We define the reward function the following way:

$$r_t = \begin{cases} +10 & \text{Core has been eliminated,} \\ +1 & \text{Core is moved farther from center,} \\ -1 & \text{All other cases,} \end{cases} \quad (9)$$

Although our final goal is to eliminate the core, we found that simply giving a reward of 10 when the goal is achieved resulted in a slow learning rate. As such, we also add a +1 reward when the core is moved towards the boundary (farther from the center). This allows our agent to get more feedback during the episode and thus learn the optimal moves faster. The -1 reward at any other step ensures that the agent tries to find the quickest way to eliminate the core.

We perform function approximation using a simple multi-layer artificial neural network (NN). We experimented with various architectures and found that a network with a single hidden layer seemed to work the best. Therefore the network has an input layer with 4 nodes (s_t), a hidden layer with 128 neurons, and finally an output layer with 9 neurons ($Q(s,a)\ \forall\, a \in \mathscr{A}$). See Fig. 2B for our NN model. The magnitude of stimulation when the stimulators are turned on is taken to be $\rho_j = \pm 50$ (with the sign determined by the learned control policy). After searching the parameter space we use the following parameters in the learning algorithm: $\gamma = 0.99$, $\alpha = 0.001$. ε decays from 1.0 to 0.1 at a rate of 0.0005 per step.

We also investigate a modified situation that considers two cores instead of one. Here, the reinforcement learning algorithm incorporates the state variable of each core as well as an extra input for each core set to 1 if that core has been pushed out of bounds and 0 otherwise. In this situation, we take the hidden layer of the neural network to include 256 neurons instead of 128. The maximum distance from the center over the course of the simulation is tracked separately for each core, and a reward is given whenever either core exceeds its previous maximum distance. Once a core reaches the edge, that core is marked as successful, and the episode succeeds if both cores are marked as successful. In this setup, the reward function is replaced with:

$$r_t = \begin{cases} +10 & \text{Both cores have moved out of bounds,} \\ +1 & \text{At least one core is farther from center,} \\ -1 & \text{All other cases,} \end{cases} \quad (10)$$

The process could similarly be extended to accommodate three or more cores.

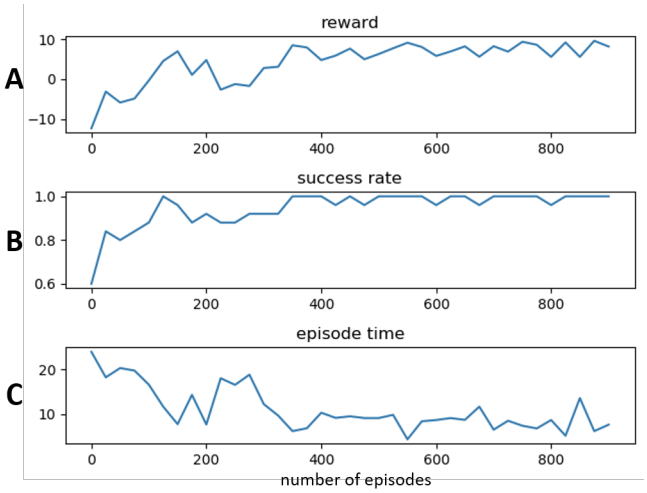

Fig. 3. Results from our single core trials. We present three different measures of the effectiveness of our reinforcement learning framework as a function of the number of episodes experienced. Each measure is the average over 25 episodes. Panel A shows the average reward achieved by our algorithm. Panel B shows the success rate, i.e., the percentages of episodes in which the cores were eliminated. Panel C shows the amount of time each episode lasts.

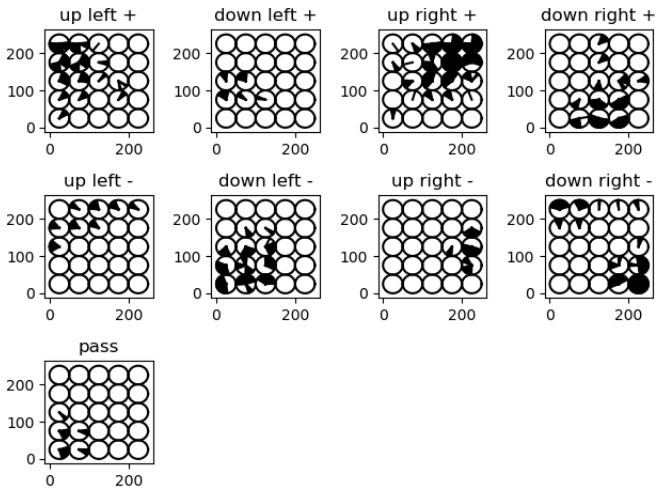

Fig. 4. A representation of actions taken at corresponding to various model states. For example, "up left +" denotes that the learned control policy would give a positive stimulus in the upper-left stimulator. "Pass" indicates that no control application is given. Each circle represents possible states (x, y, θ) of the core, where x and y correspond to the center of the circle. For each of the 9 actions, the dark segments of the circle indicate the range of θ for which that action was chosen at each position.

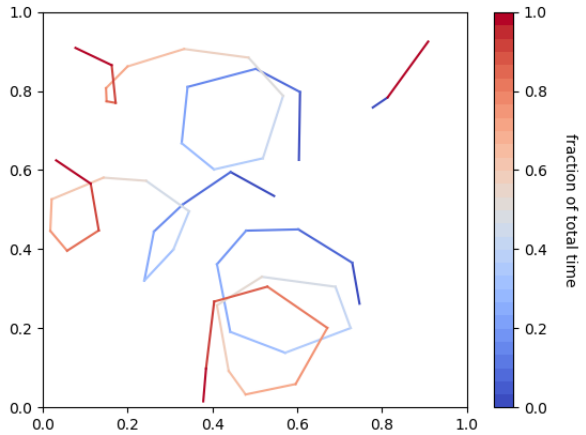

Fig. 5. Examples of trajectories from our single core trials. Each continuous line represents the motion of the core during a single episode (We present these on a single image for the sake of brevity). Blue represents the first position and red represents the last position of each core. In each of these episodes the core is successfully forced to the boundary of the domain.

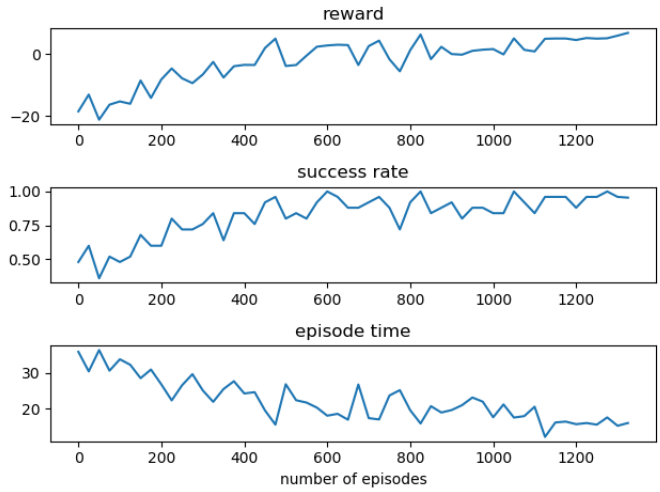

Fig. 6. Results from our double core trials. Explanations of the different measures is given in the caption of Fig. 3

IV. Results

We trained for 10,000 time units, containing approximately 900 complete episodes. Training took approximately 2 hours on a Tesla V100 GPU. For each episode, we record total reward (determined by Eq. (9)), success (whether or not the core is pushed to the edge), and total time required to complete the episode. If the learned control algorithm does not eliminate a core withing 50 time units, the trial is deemed not successful. Results during training are shown in Fig. 3. These results are an average over the last 25 episodes used in training. After training, a final evaluation over 1000 steps was performed, yielding a success rate of 100% and an average of 6.7 time units per episode.

As shown in Fig. 3B, the model starts at around a 60% success rate. That means that 60% of the cores are eliminated while taking random actions. This is reasonable given that the core starts at a random position. Therefore, when starting close to the edges of the tissue the core may simply cross the tissue boundary due to its internal dynamics or a random action. As learning progresses the success rate quickly rises until it achieves approximately 100%.

Fig. 4 gives a representation of the learned policy, representing actions taken for each state (core position and θ). We generated sample states with 25 possible core locations with θ ranging from 0 to 2π, sent these states into the learned control policy, and recorded which action was taken for each one. Because the final agent is deterministic, the same action will always be taken for a given state. Perhaps unsurprisingly, the agent tends to activate the stimulator closest to (and having the most influence on) the core; whether the activation is positive or negative depends on θ. At the same time, the policy is not completely symmetrical, and occasionally the farthest core is activated instead of the closest. Such behavior is likely a product of randomness in the training process, where multiple policies lead to a successful outcome and the model arbitrarily picks the first such policy it finds. Additionally, the learned control policy rarely takes no action, likely because we did not penalize the application of control input.

Fig. 5 shows the trajectory of the core in several episodes with different starting positions. The trajectories confirm that the model learns to push the core toward the nearest edge, even while the core is rotating due to its changing θ.

Fig. 6 shows the same metrics as in Fig. 3 for our trials with two simultaneous cores. The model steadily improves with training over 30,000 time units, corresponding to about 1300 episodes. In the final evaluation over the course of 1000 time units, the model reaches a 94.7% success rate, averaging 17.5 time units per episode. However, this result was obtained after required three times as much training as the single core model took to reach a perfect success rate. Further extension of this approach to include additional cores may be possible, but would likely require refinements to the learning algorithm.

V. Discussion and Future Work

The results presented above provide proof of concept that the reinforcement learning framework suggested here can rapidly learn strategies for control of the location of individual spiral cores using reduced order models. While initial results are promising, they only considered control of uncoupled spiral wave cores with relatively simple dynamics. In reality, fibrillation is usually caused by multiple dynamically coupled spiral waves [24], [9], [10]. As such, coupling between spiral cores with more complicated dynamical behaviors would need to be considered moving forward. Additionally, we have assumed that the underlying behavior governing fibrillation and tachycardia can be represented by a finite number of spiral cores. In practice this would likely require an accurate, real-time description of the transmembrane voltage in the heart which would be difficult to obtain in a clinical setting. Future work will investigate in situations

where only a sparse representation of the transmembrane voltage is available as might be measured near a small number of recording electrodes or using measurements from an electrocardiogram. Successful implementation in these settings would make this control framework much more likely to succeed in an implantable device.

In order to allow the reinforcement learning algorithm to accommodate more complicated model behaviors we would need to incorporate more sophisticated learning techniques. First, we would like to structure the reward in such a way that the agent is positively rewarded only when the cores are completely eliminated as opposed to our implemented reward structure from Eq. (9) which gives a reward for each step the core is moved further out. This will be necessary since this will allow the agent to learn the optimal policy without the reward biasing it towards certain actions. This reward structure would be sparse and therefore new learning techniques will be necessary.

Future work will investigate dividing our hard problem into simpler problems and then use those to help in learning the optimal policy. We can think of these simpler problems as different goals g than our final goal. In this multi-goal reinforcement learning we would have a reward function which is parameterized by the goal, r^g and an optimal policy which depends on the goal $\pi_*(s|g) = a$. These goals can be designed manually given an expertise in the subject area [25], [26] or found automatically [27], [28].

REFERENCES

[1] A. S. Go, D. Mozaffarian, V. L. Roger, E. J. Benjamin, J. D. Berry, W. B. Borden, D. M. Bravata, S. Dai, E. S. Ford, C. S. Fox, S. Franco, H. J. Fullerton, C. Gillespie, S. M. Hailpern, J. A. Heit, V. J. Howard, M. D. Huffman, B. M. Kissela, S. J. Kittner, D. T. Lackland, J. H. Lichtman, L. D. Lisabeth, D. Magid, G. M. Marcus, A. Marelli, D. B. Matchar, D. K. McGuire, E. R. Mohler, C. S. Moy, M. E. Mussolino, G. Nichol, N. P. Paynter, P. J. Schreiner, P. D. Sorlie, J. Stein, T. N. Turan, S. S. Virani, N. D. Wong, D. Woo, and M. B. Turner, "Heart disease and stroke statistics-2013 update," *Circulation*, vol. 127, no. 1, pp. e6–e245, 2013.

[2] A. Winfree, *The Geometry of Biological Time*, 2nd ed. New York: Springer Verlag, 2001.

[3] F. Fenton, E. Cherry, H. Hastings, and S. Evans, "Multiple mechanisms of spiral wave breakup in a model of cardiac electrical activity," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 12, pp. 852–892, 2002.

[4] G. P. Walcott, C. R. Killingsworth, and R. E. Ideker, "Do clinically relevant transthoracic defibrillation energies cause myocardial damage and dysfunction?" *Resuscitation*, vol. 59, no. 1, pp. 59–70, 2003.

[5] M. S. Wathen, P. J. DeGroot, M. O. Sweeney, A. J. Stark, M. F. Otterness, W. O. Adkisson, R. C. Canby, K. Khalighi, C. Machado, D. S. Rubenstein, and K. J. Volosin, "Prospective randomized multicenter trial of empirical antitachycardia pacing versus shocks for spontaneous rapid ventricular tachycardia in patients with implantable cardioverter-defibrillators pacing fast ventricular tachycardia reduces shock therapies (PainFREE Rx II) trial results," *Circulation*, vol. 110, no. 17, pp. 2591–2596, 2004.

[6] F. H. Fenton, S. Luther, E. M. Cherry, N. Otani, V. Krinsky, A. Pumir, E. Bodenschatz, and R. F. Gilmour, "Termination of atrial fibrillation using pulsed low-energy far-field stimulation," *Circulation*, vol. 120, no. 6, pp. 467–476, 2009.

[7] S. Luther, F. Fenton, B. Kornreich, A. Squires, P. Bittihn, D. Hornung, M. Zabel, J. Flanders, A. Gladuli, L. Campoy, E. Cherry, B. Luther, G. Hasenfuss, V. Krinsky, A. Pumir, R. G. Jr, and E. Bodenschatz, "Low-energy control of electrical turbulence in the heart," *Nature*, vol. 475, pp. 235–241, 2011.

[8] Y. C. Ji, I. Uzelac, N. Otani, S. Luther, R. F. G. Jr, E. M. Cherry, and F. H. Fenton, "Synchronization as a mechanism for low-energy anti-fibrillation pacing," *Heart Rhythm*, vol. 14, no. 8, pp. 1254–1262, 2017.

[9] Z. Qu, "Chaos in the genesis and maintenance of cardiac arrhythmias," *Progress in Biophysics and Molecular Biology*, vol. 105, no. 3, pp. 247–257, 2011.

[10] G. Byrne, C. D. Marcotte, and R. O. Grigoriev, "Exact coherent structures and chaotic dynamics in a model of cardiac tissue," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 25, no. 3, p. 033108, 2015.

[11] B. Rodriguez, L. Li, J. C. Eason, I. R. Efimov, and N. A. Trayanova, "Differences between left and right ventricular chamber geometry affect cardiac vulnerability to electric shocks," *Circulation Research*, vol. 97, no. 2, pp. 168–175, 2005.

[12] N. A. Trayanova, P. M. Boyle, and P. P. Nikolov, "Personalized imaging and modeling strategies for arrhythmia prevention and therapy," *Current Opinion in Biomedical Engineering*, vol. 5, pp. 21–28, 2018.

[13] V. N. Biktashev and A. V. Holden, "Resonant drift of autowave vortices in two dimensions and the effects of boundaries and inhomogeneities," *Chaos, Solitons & Fractals*, vol. 5, no. 3-4, pp. 575–622, 1995.

[14] B. Sandstede, A. Scheel, and C. Wulff, "Dynamics of spiral waves on unbounded domains using center-manifold reductions," *Journal of Differential Equations*, vol. 141, no. 1, pp. 122–149, 1997.

[15] V. S. Zykov and H. Engel, "Feedback-mediated control of spiral waves," *Physica D: Nonlinear Phenomena*, vol. 199, no. 1-2, pp. 243–263, 2004.

[16] C. D. Marcotte and R. O. Grigoriev, "Adjoint eigenfunctions of temporally recurrent single-spiral solutions in a simple model of atrial fibrillation," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 26, no. 9, p. 093107, 2016.

[17] J. Langham, I. Biktasheva, and D. Barkley, "Asymptotic dynamics of reflecting spiral waves," *Physical Review E*, vol. 90, no. 6, p. 062902, 2014.

[18] J. Langham and D. Barkley, "Non-specular reflections in a macroscopic system with wave-particle duality: Spiral waves in bounded media," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 23, no. 1, p. 013134, 2013.

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

[20] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[21] L. J. Lin, "Reinforcement learning for robots using neural networks," Carnegie-Mellon Univ Pittsburgh PA School of Computer Science, Tech. Rep., 1993.

[22] V. N. Biktashev and A. V. Holden, "Design principles of a low voltage cardiac defibrillator based on the effect of feedback resonant drift," *Journal of Theoretical Biology*, vol. 169, no. 2, pp. 101–112, 1994.

[23] Z. Cao, P. Li, H. Zhang, F. Xie, and G. Hu, "Turbulence control with local pacing and its implication in cardiac defibrillation," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 17, no. 1, p. 015107, 2007.

[24] Z. Qu, "Critical mass hypothesis revisited: role of dynamical wave stability in spontaneous termination of cardiac fibrillation," *American Journal of Physiology-Heart and Circulatory Physiology*, vol. 290, no. 1, pp. H255–H263, 2006.

[25] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, 2009, pp. 41–48.

[26] T. D. Kulkarni, K. Narasimhan, A. Saeedi, and J. Tenenbaum, "Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation," in *Advances in Neural Information Processing Systems*, 2016, pp. 3675–3683.

[27] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, P. Abbeel, and W. Zaremba, "Hindsight experience replay," in *Advances in Neural Information Processing Systems*, 2017, pp. 5048–5058.

[28] A. Graves, M. G. Bellemare, J. Menick, R. Munos, and K. Kavukcuoglu, "Automated curriculum learning for neural networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 2017, pp. 1311–1320.