Model-Based Design of Closed Loop Deep Brain Stimulation Controller using Reinforcement Learning

Qitong Gao*,1, Michael Naumann*,1, Ilija Jovanov1, Vuk Lesi1, Karthik Kamaravelu2, Warren M. Grill2, Miroslav Pajic1

1Department of Electrical and Computer Engineering, Duke University
Durham, NC USA
{*qitong.gao, michael.naumann, ilija.jovanov, vuk.lesi, miroslav.pajic*}@*duke.edu*

2Department of Biomedical Engineering, Duke University
Durham, NC USA
{*karthik.kamaravelu, warren.grill*}@*duke.edu*

Abstract— **Parkinson's disease (PD) currently influences around one million people in the US. Deep brain stimulation (DBS) is a surgical treatment for the motor symptoms of PD that delivers electrical stimulation to the basal ganglia (BG) region of the brain. Existing commercial DBS devices employ stimulation based only on fixed-frequency periodic pulses. While such periodic high-frequency DBS controllers provide effective relief of PD symptoms, they are very inefficient in terms of energy consumption, and the lifetime of these battery-operated devices is limited to 4 years. Furthermore, fixed high-frequency stimulation may have side effects, such as speech impairment. Consequently, there is a need to move beyond (1) fixed stimulation pulse controllers, and (2) 'one-size-fits-all' patient-agnostic treatments, to provide *energy efficient* and *effective* (in terms of relieving PD symptoms) DBS controllers. In this work, we introduce a deep reinforcement learning (RL)-based approach that can derive patient-specific DBS patterns that are both effective in reducing a model-based proxy for PD symptoms, as well as energy-efficient. Specifically, we model the BG regions as a Markov decision process (MDP), and define the state and action space as state of the neurons in the BG regions and the stimulation patterns, respectively. Thereafter, we define the reward functions over the state space, and the learning objective is set to maximize the accumulated reward over a finite horizon (i.e., the treatment duration), while bounding average stimulation frequency. We evaluate the performance of our methodology using a *Brain-on-Chip* (BoC) FPGA platform that implements the physiologically-relevant basal ganglia model (BGM). We show that our RL-based DBS controllers significantly outperform existing fixed frequency controllers in terms of energy efficiency (e.g., by using 70% less energy than common periodic controllers), while providing suitable reduction of model-based proxy for PD symptoms.**

I. Introduction

In 2010, Parkinson's disease (PD) affected over 680,000 individuals in the US older than 45, with projections to reach 930,000 people by 2020 [1]; this would make it the second most prevalent movement disorder [2]. Due to the number of affected individuals, PD also causes significant economic impact, with the costs in the US alone surpassing \$25B annually [3].

Deep brain stimulation (DBS) is an effective method to treat the motor symptoms of PD [4]–[7] by delivering appropriate electrical stimulation (i.e., pacing) to the basal ganglia (BG) region of the brain, as illustrated in Figure 1. Existing commercial DBS devices only feature the basic "ON-OFF" functionality, delivering fixed frequency, periodic neurostimuli. In such devices, the stimulation parameters (e.g., stimulation frequency and amplitude) are commonly hand-tuned by physicians through trial and error, remaining fixed throughout the treatment [8]. As a result, development of automated neurostimulation parameter selection methods and feedback-based stimulation controllers has attracted significant research interest in recent years.

In [9]–[14], the authors propose an *adaptive DBS* (aDBS) framework that can automatically adjust stimulation parameters based on feedback signals recorded from patients. Specifically, electromyography and accelerometer recordings, cortical neurosignals (e.g., electrocorticography), subcortical neurosignals (e.g., BG local field potentials) and neurochemical signals are used to assess patient's symptoms and to adjust neurostimulation parameters accordingly.

On the other hand, [8], [15], [16] propose the use of reinforcement learning (RL) to select appropriate pulse frequencies for standard periodic DBS to treat epilepsy; this results in reduced total stimulation energy applied to the patient's brain, and thus, increased lifetime of these battery-operated devices. Specifically, [8] proposes a method that uses EEG signals to estimate the patient's clinical states followed by minimizing the stimulation frequency and duration through RL. Based on the EEG signals, the proposed algorithm makes decisions about which stimulation frequency should be applied. In [16], a TD(0) reinforcement learning algorithm is applied to determine the stimulation frequency that minimizes stimulation energy and side effects. The authors in [15] define discrete Markov decision process (MDP) rewards over the symptoms of patients after stimulation (i.e., seizure

*Contributed equally to this work.

This work is supported in part by the CNS-1837499 and National Institutes of Health R37 NS040894 awards.

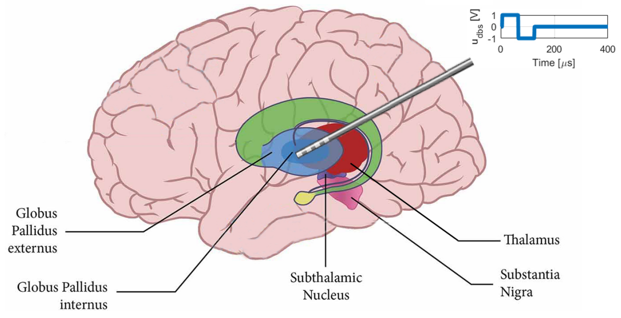


Fig. 1. Graphical depiction of deep brain stimulation (DBS) in the basal ganglia (BG). In DBS, electrodes are positioned to stimulate the subthalamic nucleus or the internal segment of the globus pallidus (`GPi`) with periodic fixed frequency trains of short-duration pulses [17].

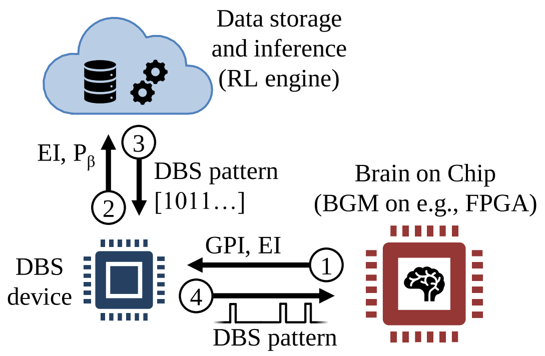

Data storage and inference (RL engine)
EI, P_β
3
DBS pattern [1011…]
2
Brain on Chip (BGM on e.g., FPGA)
DBS device
GPI, EI
1
4
DBS pattern

Fig. 2. Training dataflow pipeline: (1) quantized Error Index (EI) and globus pallidus (`GPi`) signals are acquired by the deep brain stimulation (DBS) device (and beta power spectral density P_β is computed) from the Basal Ganglia Model (BGM) implemented as the Brain on Chip (BoC) platform, and (2) EI and P_β are sent to the data storage and inference module (DSIM) which executes the reinforcement learning (RL) engine; the DSIM executes the learning algorithm, and (3) returns the corresponding DBS control policy/pattern to the DBS device, which (4) applies the DBS pattern on the BGM. In latter phases, the BGM model is replaced with live animal models.

or not-seizure), and apply the fitted-Q iteration algorithm to select suitable control actions by maximizing rewards. However, in these prior efforts, the control action space consist of simple periodic stimulation with a limited set of fixed frequencies. Their sole focus was on the search for optimal stimulation frequencies, while the stimulation pulse pattern – i.e., a simple periodic signal – remained identical across different patients.

In this paper, we introduce a deep RL-based approach for synthesis of optimal, patient-specific DBS controllers over the treatment time horizon, as summarized in Figure 2. Specifically, we consider the widely-adopted DBS performance metrics (i.e., Error Index (EI) [18] and Beta Power Spectral Density (P_β) [19]) as *Quality-of-Control* (QoC) indicators to optimize DBS controllers. To obtain an evolving (rather than static) controller policy, we model the BG region of the brains as an MDP whose state and action spaces are constituted by post-processed signals from BG regions as well as stimulation patterns, respectively; this allows DBS controllers to adapt to the pathological changes of the patient over time (e.g., to time-varying severity in the level of symptoms).

Controller adaption is supported by careful design of the corresponding reward functions to include changes in BG signals before and after stimulation. We use the evolutions of QoC metrics over time as inputs to a deep *actor-critic* (AC) learning algorithm to maximize the cumulative rewards over a finite horizon (i.e., treatment duration). The convolutional neural networks (CNNs) embedded in the AC architecture are structured specifically to extract features of the post-processed signals from the BG regions, consequently obtaining the optimal DBS control policy.

We specifically develop our algorithm on top of the AC learning framework [20], [21], since the nature of AC algorithms can effectively reduce the variance of gradient estimation and can be trained in a parallel setting [22], [23]; this results in a faster learning convergence and better control performance than the value-based RL algorithms, such as Q-learning [24] or deep Q-learning [25]. More importantly, this AC framework has been employed to solve a wide range of real-world control problems, such as automated traffic light control [26], playing video games [21], robotics [27], [28] and unmanned aerial vehicle (UAV) cooperative control [29].

To evaluate our DBS control design methodology, we develop a *Brain on Chip* (BoC) platform based on a field programmable gate array (FPGA) implementation of a physiologically-based basal ganglia model (BGM) extended from [30], [31]; the BGM was experimentally validated in live animal models (i.e., live rats), and its use for DBS device design and testing for patient conditions with varying severity of PD has already been shown in [30], [31]. The BoC enables testing of automated DBS algorithms safely and with sufficient supportive data. We experimentally show that our RL-based DBS controllers are both *energy efficient*, significantly increasing battery lifetime, AND *therapeutically effective* in terms of alleviating model-based proxies of PD symptoms; this is significant improvement over state-of-the-art periodic, fixed-frequency DBS controllers, which although successful in alleviation of PD symptoms, require more than three times the stimulation energy of our RL-based DBS controllers.

Consequently, the contributions of this work are two-fold: (1) we propose an MDP model that captures the dynamics of neuron activities in the BG by defining the state space, the action space, the transition probabilities, and the reward functions. As a result, even if the dynamics are unknown, learning algorithms can still be applied to synthesize control policies of the BGM by observing the state space and approximating the transition probabilities; and (2) we develop a deep RL algorithm for synthesis of learning-based patient-specific DBS controllers, and validate the algorithm on the BoC platform with statistically significant results.

This paper is organized as follows. Section II introduces a basal ganglia model of the brain, as well as the QoC metrics used to evaluate DBS control performance. The transformation to an MDP, enabling the reinforcement learning-

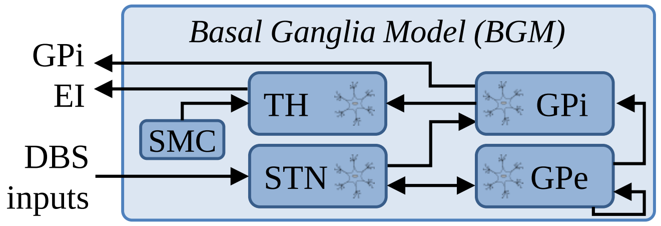


Fig. 3. Model of the brain's Basal ganglia region including *thalamic* (TH) sub-region and Sensorimotor Cortex (SMC) inputs, as well as the controller inputs for deep brain stimulation (DBS).

based approach, are detailed in Section III. Our approach is evaluated in Section IV, and possible future extensions of our work are discussed in Section V.

II. Brain Modelling and Problem Statement

In this section, we start by describing the BGM, followed by the definition of relevant QoC metrics used to assess the DBS controller performance. Additionally, PD identification from the defined QoC metrics is introduced. The BG model is presented with sufficient detail to introduce corresponding QoC metrics for controller synthesis. For an in-depth overview of the model, the reader is directed to [30], [31].

A. Computational Basal Ganglia Model

The BG region of the brain includes three sub-regions —*subthalamic nucleus* (STN), *globus pallidus pars externa* (GPe), and *globus pallidus pars interna* (GPi). Furthermore, due to its critical importance for characterizing the effects of DBS on PD symptoms (i.e., for obtaining the QoC metrics described in Section II-B), we also include the *thalamic* region (TH), as well as inputs from the *sensory motor cortex* (SMC) in the PD-relevant brain model. We collectively refer to the model of these components as the BGM, with the structure illustrated in Figure 3.

Each sub-region of the BGM is comprised of n neurons, containing a number of internal states.[1] The BGM abstracts the internal neuron's complexity by capturing its electrical potential, which is critical for modeling communication between neurons. We denote the electrical potential of a neuron as v_j^q, where $j \in \{1,..,n\}$ is the neuron's index within its sub-region, and $q \in \{TH, STN, GPe, GPi\}$ denotes the respective sub-regions. Therefore, the state of each BGM sub-region is an *electrical potential vector*

$$\mathbf{v}^q = [v_1^q \dots v_n^q]^\top. \tag{1}$$

Initial states of the neurons are model parameters that are stochastically set in our experiments. Neurons are interconnected through chemical synapses to form a BGM network (Figure 3). While these connections can be modeled as continuous dynamical systems [19], discrete-event analysis and digital implementation are facilitated by observing neuron inputs and outputs as binary events $a_j^q \in \{0,1\}$, referred to as *neural activations*, which occur when the neuron's electrical potential v_j^q exceeds a predefined threshold h_j^q. We formally capture this as

$$\begin{aligned} a_j^q(t) = 1 \Leftrightarrow & (v_j^q(t) \geq h_j^q) \wedge \\ & (\exists \delta > 0, \forall \varepsilon \in (0,\delta], v_j^q(t-\varepsilon) < h_j^q). \end{aligned} \tag{2}$$

Consequently, BGM evolution can be captured by sets of activations of all neurons over time. Formally, over arbitrary time window $[t_0, t_1]$, activations of the j^{th} neuron in the r^{th} region can be defined as a set

$$\mathcal{A}_q^j\big|_{t_0}^{t_1} = \{\tau | (t_0 \leq \tau \leq t_1) \wedge (a_j^r \text{ triggers at } \tau)\}. \tag{3}$$

Sensorimotor Cortex (SMC) inputs are embedded in TH sub-region activations, i.e., they affect the electrical potentials $v_{j\in\{1,..,n\}}^{TH}$, potentially causing activations in TH neurons. We denote the rising-edge of an SMC pulse at instant τ as $SMC_{\tau}\,\_\!\!\uparrow\!\!\!\_$. Correct activations in TH occur due to normal neural activity, while erroneous ones occur rarely in healthy patients (e.g., once in every 1000 activations). On the other hand, erroneous activations occur relatively often in patients suffering from PD (e.g., once in every few activations).

A correct activation in the TH region due to an SMC pulse is defined as

$$(\forall \tau)(\exists! t_j)(SMC_{\tau}\,\_\!\!\uparrow\!\!\!\_ \Leftrightarrow a_j^{TH}(t_j) = 1, \tau < t_j < \tau + 25ms). \tag{4}$$

In other words, an SMC pulse should trigger ***exactly one*** activation of each neuron in the TH region within 25 ms of its occurrence. An erroneous response from TH occurs after an SMC activation if the TH neuron does not respond with exactly one activation within 25 ms following an incoming SMC pulse [18].

For example, Figure 4 shows an excerpt of TH region activations due to SMC pulses; as can be seen, erroneous responses are much more likely to occur in PD brains (without DBS therapy), compared to healthy and DBS-treated PD brains when only occasionally such activations may occur (or be missed). Activations $a_{j\in\{1,..,n\}}^{TH}(t)$ over an arbitrary interval $t \in \{t_0, t_1\}$ that do not satisfy (4), form the set of erroneous activations $\mathcal{A}_{TH}^{error}\big|_{t_0}^{t_1}$, which is, as discussed in the sequel, critical for PD identification.

Recall that additional input stimuli directly applied by the DBS controller (via physically attached device probes), is also included in the model. Similarly to how SMC inputs affect the electric potentials of neurons in the TH sub-region (i.e., $v_{j\in\{1,..,n\}}^{TH}$), DBS inputs affect the electric potentials of neurons in the STN sub-region (i.e., $v_{j\in\{1,..,n\}}^{STN}$). In essence, DBS inputs influence the occurrence of neuron activations in STN, which propagates through the basal ganglia network to the TH region; here, we measure neural activity and use the acquired signals to compute QoC metrics, as will be described in the following subsection. In return, this enables assessment of the stimulation efficacy, and ultimately adjustment of neurostimulation parameters.

Finally, note that while seemingly simple, the described BGM contains all components required to extract PD-relevant QoC metrics, as we show in the rest of this section.

[1] While from the modeling perspective n is a design parameter, experiments have shown that model fidelity obtained with 10 neurons is nearly identical to models with orders of magnitude higher neuron counts [30].

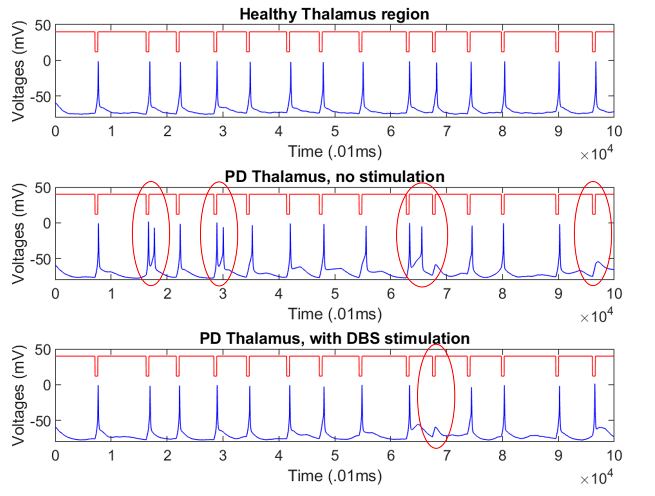


Fig. 4. (top) *Thalamic* (TH) neuron activations (in blue) of a healthy brain, (middle) a Parkinson's disease (PD) brain without deep brain stimulation (DBS), and (bottom) a PD brain with DBS treatment. Sensorimotor Cortex (SMC) pulses (highlighted in red) have been inverted and amplified to emphasize erroneous activations. The erroneous TH activations are highlighted with red ellipsoids – as can be seen, PD brains without DBS exhibit significantly higher numbers of such erroneous activations compared to a healthy brain or a PD brain with DBS.

B. *Quality of Control Metrics*

QoC for DBS controllers is determined by their efficacy in reducing PD symptoms. While multiple biomarkers (i.e., indicators of PD) are defined in literature, in the remaining of this subsection, we introduce the most commonly used Error Index (EI) [18] and Beta Power Spectral Density (P_β) [19].

1) Error Index: The Error Index (EI) is defined as

$$EI(t) = \frac{|\mathcal{A}_{TH}^{error}|_{t_0}^{t}|}{n|SMC_{\tau}\,\_\!\!\!\!\!\!\ \,\ulcorner|_{\tau=t_0}^{t}|}, t_0 = 0, \qquad (5)$$

where $|\mathcal{A}_{TH}^{error}|_{t_0}^{t}|$ is the cumulative number of erroneous responses of the TH region to the SMC pulses, and $|SMC_{\tau}\,\_\!\!\ulcorner|_{\tau=t_0}^{t}|$ is the cumulative number of SMC activations up to time t.[2] Intuitively, every neuron in TH receives the same input from SMC; hence, the number of neurons n multiplies the number of SMC activations to obtain the total number of expected activations due to SMC pulses. Note that, since EI is defined as a ratio, it is bounded to the range $[0, 1]$, and it is desired for the DBS controller to maintain EI as low as possible.

The definition (5) can be used to identify PD symptoms over relatively long periods of time, due to the included *cumulative* erroneous responses (e.g., as shown in [18]). However, as we propose changing stimulation patterns, we introduce a windowing function that enables EI computation over a sliding window of duration T_w; effectively this requires changing the static value of t_0 in (5) into $t_0 = t - T_w$,

[2]Recall that, by assumption, the SMC produces no erroneous neural activations.

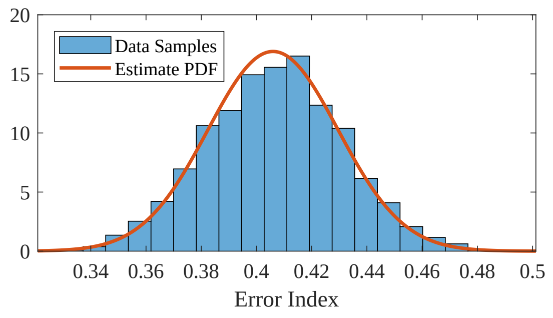


Fig. 5. Histogram of acquired windowed Error Index (EI) values estimated by a normal probability distribution function. Window size is selected as $T_w = 2s$.

resulting in (windowed) EI defined as

$$EI_{T_\omega}(t) = \frac{|\mathcal{A}_{TH}^{error}|_{t-T_w}^{t}|}{n|SMC_{\tau}\,\_\!\!\ulcorner|_{\tau=t-T_w}^{t}|}. \qquad (6)$$

The selection of time window T_w introduces a significant trade-off; reducing the time window decreases confidence in PD detection, while prolonging it may increase time-to-detection of PD symptoms. A suitable window length T_w can be determined such that the desired tradeoff is achieved. To find such T_w, we observe the variance $\sigma_{T_w}^2$ of $EI_{T_w}(t)$. For example, we considered the sliding window size $T_w = 2s$ as used for EI monitoring in [32]. To statistically validate this window size, we observed distribution of $EI_{2s}(t)$ obtained from the BGM (Figure 5). Obtained variance is $\sigma_{2s}^2 = 5.55 \cdot 10^{-4}$, providing $2\sigma_{2s}$ (95.45% confidence interval) bound of 0.047 for EI, which is sufficiently accurate for PD detection based on EI values.

2) Beta Power Spectral Density: Beta Power Spectral Density, P_β, is another PD biomarker, which we thus consider to capture DBS quality of control. In a PD brain, the GPi region exhibits pathological oscillations of neurons at frequencies within the $13\ Hz - 35\ Hz$ band (i.e., the *beta band*). Since GPi neural activity of a healthy brain does not feature the same oscillations, P_β in GPi is shown to be a suitable biomarker for PD. For example, such periodicity can be observed in Figure 6.

Figure 7 shows an excerpt of the BGM dynamics illustrating the power spectral densities of the model of a healthy brain, the model of a PD brain without DBS, and the model of the PD brain under DBS; the aforementioned pathological oscillations are especially evident in the frequency domain. Note that the P_β peak at 13 Hz is significantly more emphasized in the PD brain compared to the healthy brain or the PD brain under DBS.

We compute the beta band power for one neuron as

$$P_{\beta}{}_{j}^{GPi} = \int_{\omega=2\pi\cdot 13Hz}^{2\pi\cdot 35Hz} P_j^{GPi}(\omega)d\omega, \qquad (7)$$

where $P_j^{GPi}(\omega)$ is the single-sided power spectral density of the j^{th} neuron's potential in the GPi region. Therefore, the beta band power for the entire region with n neurons can be

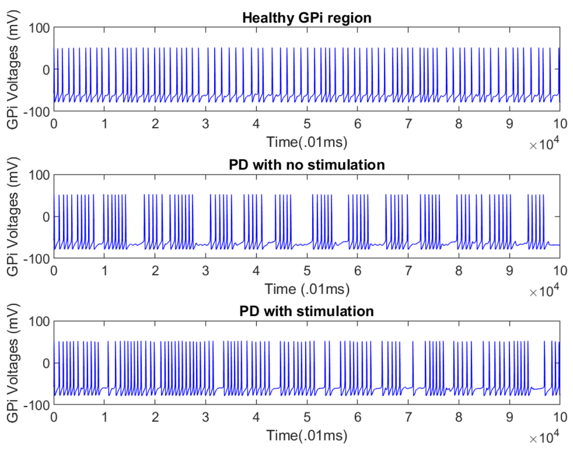


Fig. 6. Globus pallidus (GPi) activations (as highly amplified voltage signals) illustrating periodic firing patterns in a Parkinsonian GPi, versus that of healthy and stimulated Parkinson's disease (PD) (i.e., PD with DBS) brains.

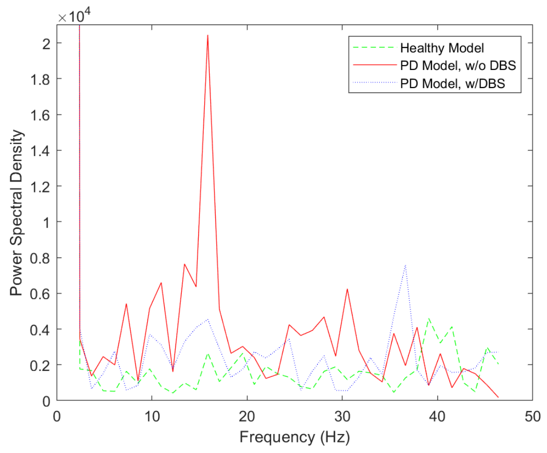


Fig. 7. Beta power spectral density P_β of globus pallidus (GPi) sub-region showing the beta band power from $13\ Hz - 35\ Hz$ band is greater for unhealthy models versus that of the healthy, and the treated brain models.

computed as

$$P_\beta^{GPi} = \frac{1}{n}\sum_{j=1}^{n} P_{\beta\,j}^{\;GPi}. \tag{8}$$

Similarly to the EI computation described in Section II-B.1, P_β^{GPi} is computed across a predefined time window.

C. Problem Statement

As discussed in Introduction, periodic high-frequency (i.e., 130-150Hz) DBS controllers are effective in alleviating PD symptoms – i.e., they provide low QoC cost (low EI and P_β levels). However, to achieve this, they are also very energy inefficient, due to the high stimulation rates (frequency). Furthermore, by applying 'one-size-fits-all' periodic stimulation, they don not support patient-specific effective DBS that could additionally improve energy efficiency; to obtain patient-specific stimulation policies, it is necessary to develop learning-based DBS controllers.

Consequently, our objective is to derive *learning-based methods* for design of DBS controllers that are both *energy efficient* and *therapy effective* (i.e., with a desired (low) QoC cost). Given the PD-relevant Basal Ganglia model described in Section II-A and the relevant DBS controller QoC metrics defined in Section II-B, we are able to map this problem into the following two essential challenges.

- How can we transform the computational BG model into a form suitable for efficient stimulation pattern exploration, given the large state/action (i.e., biomarker/stimulation pattern) spaces?
- How can we formulate a problem for synthesis of QoC-optimal DBS control policies (i.e., stimulation patterns) in a way that facilitates learning-based DBS design?

In what follows, we address these challenges by introducing a suitable transformation from the computational BGM to an MDP-based model; such MDP model is then utilized as a foundation for our Reinforcement Learning (RL)-based synthesis of DBS patterns that minimize energy consumption while providing the desired QoC level.

III. Reinforcement Learning Formulation of the DBS Controller Synthesis Problem

In this section, we start by introducing MDPs and actor-critic (AC) learning, after which we define the transformation of the BGM (described in Section II-A) into an MDP-compliant representation. Finally, we introduce a deep RL algorithm for design of effective and energy efficient DBS patterns (i.e., controllers).

A. MDPs and Actor-Critic Learning

We start with the following definitions.

Definition 3.1 (MDP): An MDP is a tuple $\mathcal{M} = (\mathcal{S}, s_0, \mathcal{U}, \mathcal{P}, R, \gamma)$, where $\mathcal{S}$ is a finite set of states; s_0 is the initial state; $\mathcal{U}$ is a finite set of actions; P is the transition probability function defined as $\mathcal{P} : \mathcal{S} \times \mathcal{U} \times \mathcal{S} \to [0,1]$; $R : \mathcal{S} \times \mathcal{U} \times \mathcal{S} \to \mathbb{R}$ is the reward function, and $\gamma \in [0,1]$ is a discount factor.

Definition 3.2 (Control Policy of MDPs): A policy π of an MDP $\mathcal{M}$ is a function, $\pi : \mathcal{S} \to \mathcal{U}$, that maps the set of states $\mathcal{S}$ to the set of (control) actions $\mathcal{U}$.

The goal of RL is to find an optimal (control) policy π^* that maximizes the accumulated return, which is defined as follows.

Definition 3.3 (Accumulated Return): Given an MDP $\mathcal{M}$ and a policy π, the accumulated return over a finite horizon starting from the stage t and ending at stage $t+T$, for $T > 0$, is defined as $G_t = \sum_{k=0}^{T} \gamma^k r_{t+k}$, where r_{t+k} is the return at the stage $t+k$.

Now, we formally introduce the objective of RL – given an MDP $\mathcal{M}$ with ***unknown*** transition probabilities P and a pre-defined reward function R, find a policy π^* such that the expected accumulative return starting from the initial stage over the entire horizon is maximized; this is formally captured as

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}_{s_t \in \mathcal{S}, u_t \sim \pi(s_t), r_t \sim R}[G_1], \tag{9}$$

where $s_t \in S$ denotes that the states $s_1, s_2, \ldots, s_T$ at each stage are selected from the finite state set $\mathcal{S}$, $u_t \sim \pi(s_t)$ denotes that the actions u_t at any stage t are determined by the policy π, and $r_t \sim R$ denotes that the rewards at any stage t are determined by the reward function R.

The optimization problem in (9) has been extensively studied, and various algorithms proposed (e.g., [20]–[22], [25], [33]–[35]), which can be used to solve the DBS control problem that we are interested in. We employ the AC learning framework [20], [21] as AC-based algorithms result in a reduced variance of gradient estimation, faster learning convergence and better control performance than the value-based RL algorithms [22], [23]. We now introduce the essential components of AC-based learning; further theoretical details can be found in [20], [21], [24] and references therein. We start by defining the state-action value functions.

Definition 3.4 (State-Action Value Function): Given an MDP $\mathcal{M}$ and policy π, the state-action value function $Q^\pi(s,u)$, where $s \in \mathcal{S}$ and $u \in \mathcal{U}$, is defined as the expected return for taking action u when at state s following policy π at stage t, i.e.,

$$Q^\pi(s,u) = \mathbb{E}_{s \sim \mathcal{S}, u \sim \mathcal{U}}[G_t | s_t = s, u_t = u]. \tag{10}$$

The major components of AC-based learning problems are the *actor* network $\mu(s|\theta_u) : \mathcal{S} \to \mathcal{U}$ and the *critic* network $Q(s,u|\theta_c) : \mathcal{S} \times \mathcal{U} \to \mathbb{R}$, where a neural network (NN) with weights θ_u is used to approximate the optimal policy π^* and another NN with weights θ_c is used to approximate $Q(\cdot,\cdot)$. During training, the two networks are updated sequentially (i.e., one after the other), and as the training evolves, the actor network $\mu(s|\theta_u)$ should converge to π^*. Specifically, in each training epoch, θ_c first performs a gradient descent step $\theta_c \leftarrow \theta_c - \alpha_c \nabla_{\theta_c} J_c(\theta_c)$ towards minimizing the critic network loss function $J_c(\theta_c)$, which is defined as

$$\begin{aligned} J_c(\theta_c) =& \mathbb{E}_{s_t, s_{t+1} \sim \delta^\mu, u_t \sim \mu, r \sim R}[(r + \\ & \gamma Q(s_{t+1}, \mu(s_{t+1}|\theta_u)|\theta_c) - Q(s_t, u_t|\theta_c))^2]. \end{aligned} \tag{11}$$

Here, δ^μ is the state visitation distribution over the exploration policy μ, and $s_t \in \mathcal{S}$ is drawn from this distribution, action u_t follows the exploration policy μ, the reward r is acquired following the reward function R, and α_c is the learning rate of the critic network.

In next step, the θ_u performs a gradient ascent step $\theta_u \leftarrow \theta_u + \alpha_u \nabla_{\theta_u} J_u(\theta_u)$, where

$$\nabla_{\theta_a} J_u(\theta_u) = \mathbb{E}_{s_t \sim \delta^\mu}[\nabla_{\theta_u} Q(s_t, \mu(s_t|\theta_u)|\theta_c)], \tag{12}$$

and α_u is the learning rate of the actor network.

B. BGM to MDP Transformation

In this section, we formulate an MDP which captures the dynamics of the neuron activities of the BGM. We first define the state space $\mathcal{S}$, action space $\mathcal{U}$, and the reward function R that are used to formulate the RL problem. We define the state $s_m \in \mathcal{S}$, $m \geq 0$, as a matrix in $\mathbb{R}^{2\times l}$ whose first row is a discretized sequence of past EI signals and second row is a discretized sequence of past P_β signals, respectively, starting from the time step $m \cdot l$ to $(m+1)\cdot l$; here, $l \in \mathbb{R}$ is the period of sampling, such that $l \in (0, T/2]$ where T denotes the entire time-horizon of the stimulation for one round of patient treatment, and $m \in \mathbb{Z}_0^+$. Formally, this is captured as

$$\begin{aligned} s_j &= \left[s_{m_{(0)}}, s_{m_{(1)}}, s_{m_{(2)}}, \ldots, s_{m_{(l-1)}}\right] \\ &= \begin{bmatrix} e_{m_{(0)}}, e_{m_{(1)}}, e_{m_{(2)}}, \ldots, e_{m_{(l-1)}} \\ \beta_{m_{(0)}}, \beta_{m_{(1)}}, \beta_{m_{(2)}}, \ldots, \beta_{m_{(l-1)}} \end{bmatrix}, \end{aligned} \tag{13}$$

where the $e_{m_{(i)}}$ in the first row refer to the EI at time step $m_{(i)}$, $\beta_{m_{(i)}}$ in the second row refer to the P_β at time step $m_{(i)}$, and $m_{(i)} = m \cdot l + i$, for all $i \in [0, l-1]$. The action $u_m \in \mathcal{U}$ starting from the time step $m \cdot l$ to $(m+1)\cdot l$ is represented by a vector in $\mathbb{R}^l$, i.e.,

$$u_m = [u_{m_{(0)}}, u_{m_{(1)}}, u_{m_{(2)}}, \ldots, u_{m_{(l-1)}}], \tag{14}$$

where for all $i \in [0, l-1]$ it holds that

$$u_{m_{(i)}} = \begin{cases} 1, & \text{if a pulse is placed at time step } t_i \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

Finally, we define the reward function as

$$R(s_m, u_t, s_{m+1}) = \begin{cases} r_a, & \text{if } \bar{e}_{m+1} < \mathcal{T}_e \text{ and } \bar{\beta}_{m+1} < \mathcal{T}_\beta \\ r_b, & \text{if } \bar{e}_{m+1} \geq \mathcal{T}_e \text{ and } \bar{\beta}_{m+1} < \mathcal{T}_\beta \\ r_b, & \text{if } \bar{e}_{m+1} < \mathcal{T}_e \text{ and } \bar{\beta}_{m+1} \geq \mathcal{T}_\beta \\ r_c, & \text{if } \bar{e}_{m+1} \geq \mathcal{T}_e \text{ and } \bar{\beta}_{m+1} \geq \mathcal{T}_\beta \end{cases} \tag{16}$$

where $\bar{e}_{m+1} = \frac{1}{l}\sum_{k=0}^{l} e_{m+1_{(k)}}$, $\bar{\beta}_{m+1} = \frac{1}{l}\sum_{k=0}^{l} \beta_{m+1_{(k)}}$, $\mathcal{T}_e \in \mathbb{R}$, $\mathcal{T}_\beta \in \mathbb{R}$, and $r_a >> r_b > 0 > r_c$. Intuitively, we assign a large positive reward r_a to the states whose averages of both EI and P_β are below the threshold $\mathcal{T}_e$ and $\mathcal{T}_\beta$ after stimulation, respectively. A small positive reward r_b is assigned to the states which only one of the $\bar{e}_{m+1}$ or $\bar{\beta}_{m+1}$ value is below its corresponding threshold; otherwise, a negative reward r_c with a small absolute value is assigned to all the other states. At last, the transition probabilities $\mathcal{P}$ are captured by initial states of the electrical potential vector $\mathbf{v}$ from (1), since it is the source of stochasticity in the BGM [31], and in our case are considered unknown.

C. Deep RL Algorithm for Stimulation Pattern Synthesis

We now introduce the learning objective for solving the DBS controller problem stated in Section II-C; i.e., that minimizes QoC cost while limiting energy consumption (captured as average stimuli freqency). We then present a method to solve this learning problem.

Problem 1 (Learning Objective): Given an MDP $\mathcal{M}$ that represents the BGM's neural activity with unknown transition probabilities $\mathcal{P}$, and a stimulation frequency f that is lower than the maximum stimulation energy specified in terms of frequency f_{max} (i.e. $f < f_{max}$), find the optimal policy π^* that maximizes the expected accumulated return starting from the initial stage, as defined in (9).

Algorithm 1 Policy generation through deep AC learning

```
Input: Frequency of pulses $f < f_{max}$ in the actions $u_m$
Begin:
 1: Initialize the actor network $\mu(\cdot|\theta_u)$ with $f$
 2: Initialize the critic network $Q(\cdot,\cdot|\theta_c)$
 3: for $epi = 0$ to $max\_epi$ do
 4:     Reinitialize the neuron activities with a new initial state and collect from the brain model EI and $P_\beta$ with duration $l$ and form the initial state $s_0$ following (13)
 5:     $s \leftarrow s_0$
 6:     for $j = 0$ to $max\_epoch - 1$ do
 7:         $u \leftarrow \mu(s|\theta_u) + \mathcal{N}$ ▷ Evaluate the actor network
 8:         Stimulate $a$ for duration $l$. In the mean time, collect from the brain EI and $P_\beta$ with duration $l$ and form $s'$ following (13)
 9:         Calculate reward $r = R(s,u,s')$ following (16)
10:         Append tuple $(s,u,r,s')$ to the training buffer $\mathcal{B}$
11:         Sample minibatch $\tilde{\mathcal{B}}$ uniformly from $\mathcal{B}$
12:         Update critic network $\theta_c \leftarrow \theta_c - \alpha_c \nabla_{\theta_c} J_c^{\tilde{\mathcal{B}}}(\theta_c)$
13:         Update actor network $\theta_u \leftarrow \theta_u + \alpha_u \nabla_{\theta_u} J_u^{\tilde{\mathcal{B}}}(\theta_u)$
14:         $s \leftarrow s'$
15:     end for
16: end for
17: $\pi^*(\cdot) \leftarrow \mu(\cdot|\theta_u)$
18: return $\pi$
```

To solve Problem 1, we start by devising the dataflow pipeline illustrated in Fig. 2. First, quantized EI and GPi signals are acquired from the the BoC platform (implementing the BGM) by the DBS device, where the P_β biomarker is computed, and, along with EI, forwarded to the data storage and inference module (DSIM) – i.e., a computing platform on the edge or in the cloud where the RL engine is executed. The DSIM postprocesses the signals and executes the learning algorithm Alg. 1; in return, the corresponding DBS stimulation pattern is synthesized and sent to the DBS device, which then applies the DBS pattern on the BGM. Note that the EI and P_β signals constitute the state $s \in \mathcal{S}$ and the stimulation patterns constitute the actions $u \in \mathcal{U}$ in the MDP $\mathcal{M}$, respectively. While the actual brain with implanted probes replaces the BoC platform in the real operating environment, BoC allows evaluation of the learning algorithm as it can generate relevant signals in real-time (further discussion is provided in Section V).

Alg. 1 processes the discretized EI and P_β signals and learn the optimal control policy – i.e., a sequence of brain stimulation pulses that reduces effects of PD by minimizing the QoC cost. PD symptoms are identified from the discretized EI and P_β signals, where higher EI and P_β values signify that the patient is highly affected by PD (as described in Section II-B). Specifically, Alg. 1 takes input f, which defines the average frequency of the stimulation controller, and initializes the actor network and the critic networks (i.e., $\mu(\cdot|\theta_u)$ and $Q(\cdot,\cdot|\theta_c)$, respectively). The inputs of the actor network $\mu(\cdot|\theta_u)$ are states while outputs are actions; the critic network takes as input both states and actions and in return computes the predicted state-action values of the inputs.

Since we need to process real-time signals (i.e., time series), we use CNNs to extract the hidden features and to approximate the desired outputs. CNNs have been proven effective in image analysis and natural language processing [36]. However, in most cases the inputs to the CNNs are time-independent (e.g., as in images), while in our case, the inputs to the CNNs are time series.

Although some work has been done on the use of CNNs on time-indexed data [37]–[39], existing methods mostly focus on classification problems instead of regression, which deviates from our objective. Also, considering that the state space is represented by two correlated time sequences, which is unique to the existing works, in this paper we design a CNN structure specifically for the actor network and the critic network. Furthermore, in our case the action vectors u_m contains finite elements which are only 0s and 1s, consequently, and thus we design a special actor network structure whose outputs meet this requirement.

1) Actor-Critic Networks Design: In the actor network, the inputs in $\mathbb{R}^{l\times 2}$ are first convoluted with 32 convolution filters (each with dimension 1×3), followed by an average pooling layer (with dim. 1×4) to reduce the overall feature map dimension. The output of this layer is in $\mathbb{R}^{l/4\times 32}$. Then, 64 convolution filters (each with size 1×3), are convoluted with the outputs from the previous average pooling layer, and the resulting feature maps are processed by an additional average pooling layer (with dim. 1×4), which leads to outputs in $\mathbb{R}^{l/16\times 64}$. Then, all the feature vectors from the previous average pooling layer are sequentially flattened and concatenated; finally, two fully connected layers, (each with 1000 nodes), are used to map the resulting feature vectors to the *action logits* $u_m^{logits} \in \mathbb{R}^{b\cdot l}$, where b represents the number of non-zero elements in the action u_m. Given f in Hz and l in ms, b can be obtained as $b = f\cdot l/1000$.

Action logits u_m^{logits} can be translated to the actions u_m as follows. First, the u_m^{logits} is reshaped to a matrix u_m^{mat} in $\mathbb{R}^{b\times l}$ and the Softmax function

$$\sigma(\mathbf{x})_i = \frac{e^{x_i}}{\sum_{k=0}^{K-1} x_k},\ \mathbf{x}\in\mathbb{R}^K, \tag{17}$$

is applied to each row entry of u_m^{mat} to normalize the summation of each row to 1, where we denote the output matrix from the Softmax as $u_m^p \in \mathbb{R}^{b\times l}$. We have that

$$\begin{aligned} u_m^p\{y,z\} &= \frac{u_m^{mat}\{y,z\}}{\sum_{k=0}^{l-1} u_m^{mat}\{y,k\}} \\ &= \frac{u_m^{logits}\{y\cdot b+z\}}{\sum_{k=0}^{l-1} u_m^{logits}\{y\cdot b+k\}}, \end{aligned} \tag{18}$$

where $\sum_{k=0}^{l-1} u_m^p\{y,k\} = 1$ and $u_m^{\cdot}\{y,z\}$ refers to the element located at the yth row and the zth column if $u_m^{\cdot}$ is a matrix, and with a little abuse of notations, $u_m^{\cdot}\{y\}$ refers to the yth element of $u_m^{\cdot}$ if it is a vector and all indices start from 0. Note that $u_m^p\{y,z\}$ are expected to represent the probability that the ith pulse, where $i \in [0, b-1]$, appears at the index i in the action vector u_m, i.e., $u_{m_{(i)}} = 1$.

In what follows, the maximum element of each row of u^p_m are selected to form the pulse index vector u^{idx}_m, where $u^{idx}_m\{i\} = \max u^p_m\{i,:\}\ \forall i \in [0, b-1]$ and $u^p_m\{i,:\}$ refers to all the elements in the ith row of matrix u^p_m. Finally, the elements in the action vector u_m are determined by

$$u_{m_{(i)}} = \begin{cases} 1 & \text{if } i \in u^{idx}_m, \\ 0 & \text{otherwise}, \end{cases} \tag{19}$$

where $i \in [0, l-1]$.

The critic network shares the same convolutional and fully connected layers with the actor network, while the action logits vector u^{logits}_m is included at the last fully connected layer. The structure of the actor network and the critic network are shown in Fig. 8 and 9, respectively.

2) AC Network Training: In Alg. 1, after the initialization (Alg. 1, lines 1-2), we start each training episode by initializing the brain model with a new initial state of neuron electric potentials (via vector $\mathbf{v}$, as defined in (1)), with a different set of potentials. Note that this step ensures sufficient exploration of the MDP's state space. Then, by acquiring and collecting the EI and P_β signals from the BGM over the duration l, the current MDP state s is constructed as defined in (13) (Alg. 1, lines 4-5). Finally, in each training epoch, the action u, representing the stimulation pattern defined in (14), is calculated by superimposing random noise $\mathcal{N}$ on the output from the actor network $\mu(s|\theta_u)$ (Alg. 1, line 8). Here, the noise is used to ensure sufficient exploration of the state space during learning [21].

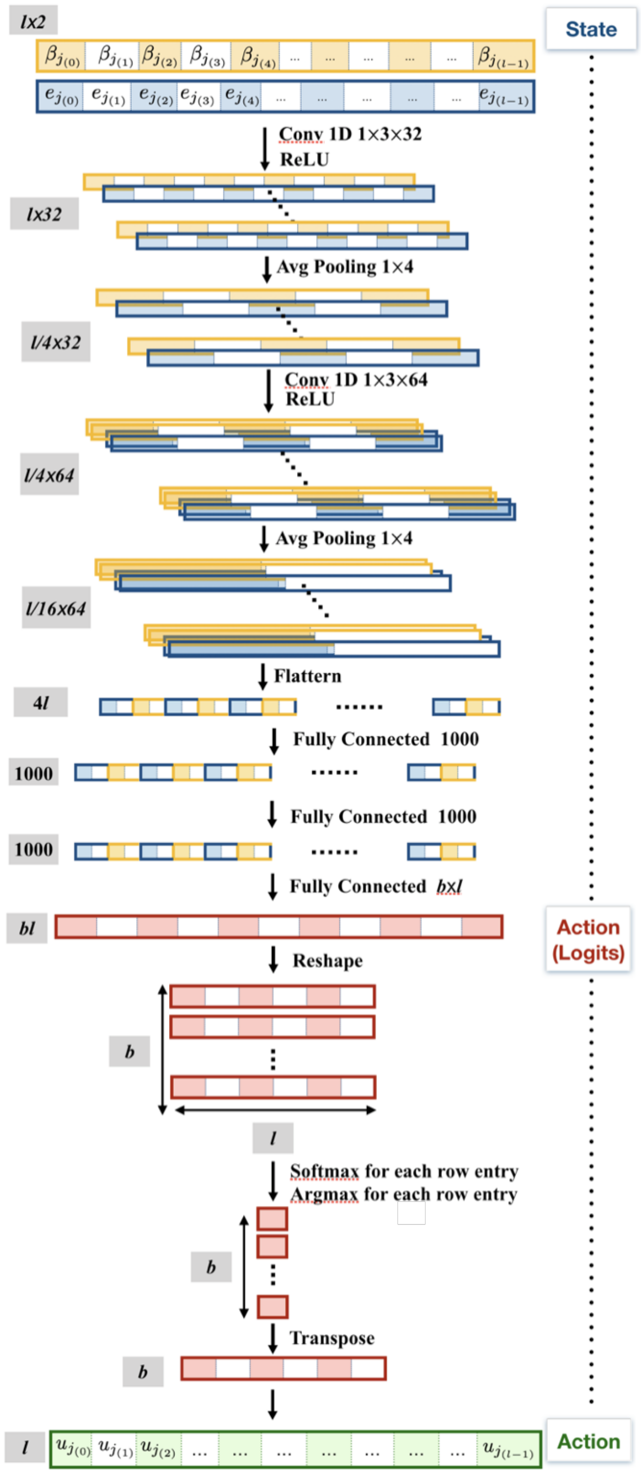
The obtained stimulation pattern is applied the BG model through through the DBS device; as a result, a new window of EI and P_β sequences, each with duration l, are acquired by the DBS device and transmitted to the DSIM; then the two sequences are assigned to s' (Alg. 1, line 8). The AC-learning step is performed (Alg. 1, line 9-13) by first calculating the reward r as in (23), and then updating the critic and actor networks respectively. We apply batch training [40] to accelerate the training process and reduce variance of the gradients (11) and (12). Specifically, we define the empirical experience buffer to store all the empirical data that has been encountered until time step t as

$$\mathcal{B} = \{(s^{(t)}, u^{(t)}, s^{(t+1)}, r^{(t)}) | t \in [0, T]\}, \tag{20}$$

where $(s^{(t)}, u^{(t)}, s^{(t+1)}, r^{(t)})$ is the learning experience at time step t. During training, minibatches of experiences $\tilde{\mathcal{B}} = \{(s^{(t)}, a^{(t)}, s^{(t+1)}, r^{(t)})\}_{\mathcal{B}}$ are drawn uniformly from the buffer $\mathcal{B}$ to calculate the gradients of the *batch* cost function

$$\begin{aligned}\nabla_{\theta_c} J^{\tilde{\mathcal{B}}}_c(\theta_c) = &\nabla_{\theta_c} \mathbb{E}_{\{(s^{(t)}, a^{(t)}, s^{(t+1)}, r^{(t)})\}_{\tilde{\mathcal{B}}}} [(r + \\ &\gamma Q(s^{(t+1)}, \mu(s^{(t+1)}|\theta_u)|\theta_c) - Q(s^{(t)}, u^{(t)}|\theta_c))^2],\end{aligned} \tag{21}$$

$$\nabla_{\theta_u} J^{\tilde{\mathcal{B}}}_u(\theta_u) = \frac{1}{n} \sum_{s^{(t)} \in \tilde{\mathcal{B}}} \mathbb{E}_{s^{(t)} \sim \delta^{\tilde{\mathcal{B}}}} [\nabla_{\theta_u} Q(s^{(t)}, \mu(s^{(t)}|\theta_u)|\theta_c)]. \tag{22}$$



lx2
State
$\beta_{j(0)}$ $\beta_{j(1)}$ $\beta_{j(2)}$ $\beta_{j(3)}$ $\beta_{j(4)}$ ... $\beta_{j(l-1)}$
$e_{j(0)}$ $e_{j(1)}$ $e_{j(2)}$ $e_{j(3)}$ $e_{j(4)}$ ... $e_{j(l-1)}$
Conv 1D 1×3×32
ReLU
lx32
Avg Pooling 1×4
l/4x32
Conv 1D 1×3×64
ReLU
l/4x64
Avg Pooling 1×4
l/16x64
Flatten
4l
Fully Connected 1000
1000
Fully Connected 1000
1000
Fully Connected bxl
bl
Action (Logits)
Reshape
b
l
Softmax for each row entry
Argmax for each row entry
b
Transpose
b
l
$u_{j(0)}$ $u_{j(1)}$ $u_{j(2)}$ ... $u_{j(l-1)}$
Action

Fig. 8. Structure of the actor network $\mu(\cdot|\theta_u)$.

As the final step, the critic and the actor networks are updated according to (21) and (22) (Alg. 1, lines 12-13). At the end of each training epoch, the current state s is replaced by s' (Alg. 1, line 14), and another training episode is initiated by re-initializing the BGM with a set of random initial neuron potential conditions. After training, the actor the network gives the QoC-optimal stimulation pattern for

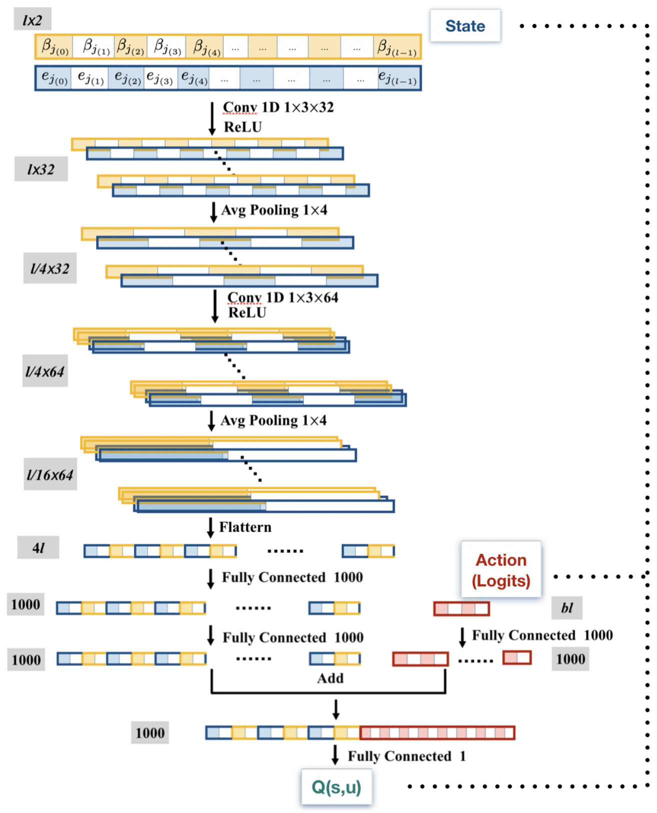


Fig. 9. Structure of the critic network $Q(\cdot,\cdot|\theta_c)$.

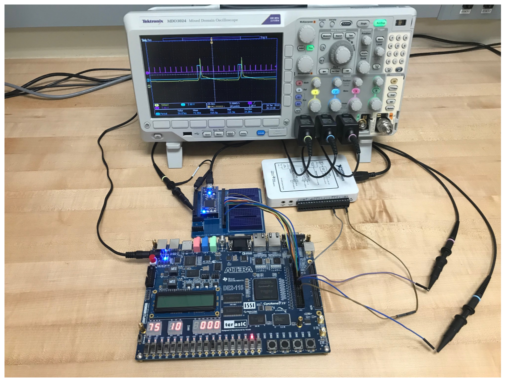


Fig. 10. Experimental setup for deep brain stimulation (DBS) controller evaluation based on the basal ganglia model (BGM) hardware platform.

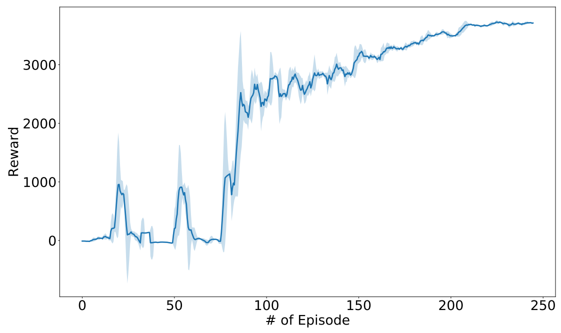


Fig. 11. Accumulated reward of each training episode – Alg. 1 starts converging after 80 episodes of training and finds the optimal stimulation control policy after 200 episodes of training.

all the states $s \in \mathcal{S}$, and serves as the optimal policy π^* (Alg. 1, line 17).

IV. Evaluation of RL-Based DBS Controllers

In this section, we evaluate the RL-based DBS controllers derived by Alg. 1; to achieve this we utilize the BoC platform, shown in Figure 10, which implements the BGM on Altera DE2-115 FPGA board and acts as a safe testbed for DBS controllers design and analysis. Furthermore, we compare the performance of the derived RL-based DBS controllers against the periodic stimulation patterns that are employed in [8], [15], [16].

In Alg. 1, we selected the sampling duration $l = T_w = 2sec$, stimulation frequency $f = 45$ Hz, maximum number of training episode $max\_epi = 250$, the maximum step in each training episode $max\_step = 50$, actor learning rate $\alpha_u = 0.001$ and critic learning rate $\alpha_c = 0.01$. Figure 11 shows the accumulated reward obtained in each training episode. Observe that Alg. 1 starts to converge from around 80th episode and finds the optimal control pattern after around 200 episodes of training, for the reward function from (16) that was specifically defined as

$$R(\cdot,\cdot,\cdot) = \begin{cases} 100, & \text{if } \bar{e}_{m+1} < 0.1 \text{ and } \bar{\beta}_{m+1} < 25\cdot 10^5 \\ 1, & \text{if } \bar{e}_{m+1} \geq 0.1 \text{ and } \bar{\beta}_{m+1} < 25\cdot 10^5 \\ 1, & \text{if } \bar{e}_{m+1} < 0.1 \text{ and } \bar{\beta}_{m+1} \geq 25\cdot 10^5 \\ -1, & \text{if } \bar{e}_{m+1} \geq 0.1 \text{ and } \bar{\beta}_{m+1} \geq 25\cdot 10^5 \end{cases} \tag{23}$$

Furthermore, we compared the performance of the derived RL-based DBS controller, which has an average stimulation frequency of 45 Hz, with the performance of the 45 Hz and 180 Hz periodic controllers, for which the stimulation pulses are equally spaced in terms of time steps. Figure 12, Figure 13 and Figure 14 present the changes of the EI, P_β and `TH` neuron activations in a 10s window respectively, when the DBS controllers are not activated until time 2000ms, and remain on for the rest of the 10s test window.

In Figure 12 and Figure 13, the blue line shows the performance of the RL-based controller, the orange line shows the performance of the 45 Hz periodic controller and the green line shows the performance of the 180 Hz periodic controller. By comparing with the 45 Hz periodic controller, which has the same frequency as the RL-based controller derived by Alg. 1, it can be observed that the RL controller significantly reduces both the EI and the P_β; thus, providing effective PD therapy. In addition, although the 180 Hz periodic controller can further reduce the EI and P_β, the RL controllers provides suitable QoC cost (i.e., it alleviates PD symptoms) while reducing the stimulation frequency by 75%; hence, significantly increasing lifetime of these battery operated devices.

Finally, in Figure 14, we observe that stimulating with either the RL controller derived using Alg. 1 or the 180 Hz

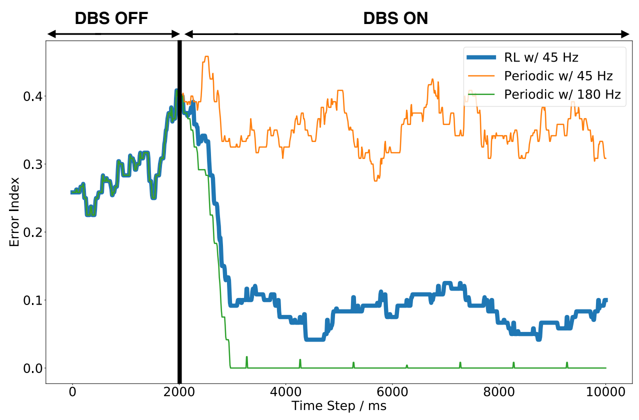


Fig. 12. Error Index (EI) over time after stimulating with different deep brain stimulation (DBS) controllers. DBS is off in the first 2000ms and on for the rest of testing period. Blue line shows the obtained results when the derived RL-based controller is used. Orange and green lines show the results when classical periodic controllers with 45 Hz and 180 Hz, respectively, were utilized.

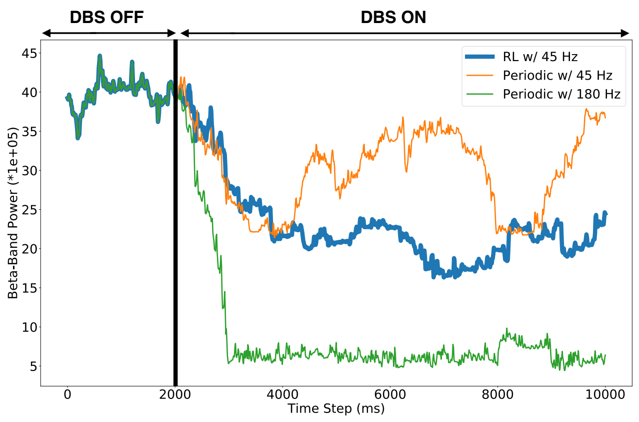


Fig. 13. Beta power spectral density P_{β} over time after stimulating with different patterns. Deep brain stimulation (DBS) is off in the first 2000ms and then on for the rest of the testing period. Blue line shows the obtained results for the derived RL-based controlled. Orange and green lines show the results when classical periodic DBS controllers with 45 Hz and 180 Hz, respectively, were used.

periodic controllers stabilizea the TH activations, effectively alleviating the symptoms of the PD, as discussed in Section II-A. On the other hand, the 45 Hz periodic controllers cannot achieve such therapy effects. Thus, the RL DBS controllers is capable of providing the desired PD therapy effects with less than a third of the energy cost of commonly used DBS controllers.

V. Discussion and Conclusion

Most existing commercial Deep Brain Stimulation (DBS) devices only employ periodic stimulation with fixed high-frequency stimulation, which is tuned by physicians through trial-and-error. As a result, these devices can either achieve therapy effectiveness OR energy efficiency. To address this major limitation, in this work, we have introduced a reinforcement learning (RL)-based method for design of DBS controllers that are both therapeutically effective, in terms of alleviating Parkinson Disease (PD) symptoms, AND energy efficient, by significantly increasing lifetime of these battery-operated devices. To achieve this, we have derived a procedure to map automatically a Basal Ganglia (BG) model, capturing neural activity in the related brain regions, into a Markov Decision Process (MDP)-based model; within the MDP model, the state and action space have been defined as signals gathered from the BG sub-regions and the stimulation patterns, respectively. Finally, the learning objective has been set to maximize the accumulated reward over a finite time-horizon.

We have evaluated the performance of our RL-based DBS controllers using a Brain-on-a-Chip (BoC) hardware platform that encodes the previously validated Basal Ganglia Model (BGM). We have shown that our method significantly outperforms existing, commonly used, periodic DBS controllers. We have presented a novel approach that effectively alleviates PD symptoms while finding optimal DBS control patterns, with minimal stimulation frequency, through the use of reinforcement learning.

In an effort to achieve both higher energy efficiency of DBS controllers as well as therapeutic effectiveness in alleviating PD symptoms, our findings suggest that generating temporal patterns is important for enhancing DBS treatment, as the RL-based DBS controllers significantly outperformed the periodic controllers with the same frequency. The RL approaches from [8], [15], [16] found that varying fixed frequencies over treatment can improve efficacy; our results suggest that even higher efficacy can be achieved at those frequencies with optimal temporal patterns.

The framework proposed herein can also be employed to develop patient specific controllers for use in human or animal testing. Specifically, we are in the process of replacing the BoC platform from Figure 2 with preclinical (live) animal models. The results from this work suggest that optimizations over the frequency domains is possible, and combined with the work similar to [8], [15], [16], could result in higher efficacy in controllers that can improve the level of DBS treatment.

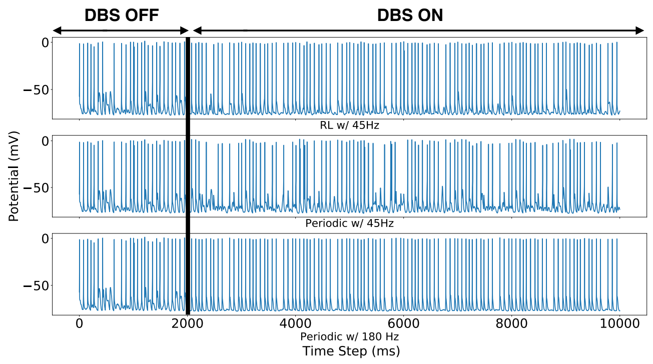


Fig. 14. Thalamic (TH) neuron activity after stimulating with different DBS patterns. Obtained results when RL-based controller is used are shown in the top row, while results when classical periodic controllers with frequency of 45 Hz and 180 Hz are utilized are shown in the second and third row, respectively.

References

[1] C. Marras, J. Beck, J. Bower, E. Roberts, B. Ritz, G. Ross, R. Abbott, R. Savica, S. Van Den Eeden, A. Willis, *et al.*, "Prevalence of parkinson's disease across north america," *NPJ Parkinson's disease*, vol. 4, no. 1, p. 21, 2018.

[2] O.-B. Tysnes and A. Storstein, "Epidemiology of parkinson's disease," *Journal of Neural Transmission*, vol. 124, no. 8, pp. 901–905, 2017.

[3] R. L. Cutler, F. Fernandez-Llimos, M. Frommer, C. Benrimoj, and V. Garcia-Cardenas, "Economic impact of medication non-adherence by disease groups: a systematic review," *BMJ open*, vol. 8, no. 1, p. e016982, 2018.

[4] A. L. Benabid, "Deep brain stimulation for parkinson's disease," *Current opinion in neurobiology*, vol. 13, no. 6, pp. 696–706, 2003.

[5] G. Deuschl, C. Schade-Brittinger, P. Krack, J. Volkmann, H. Schäfer, K. Bötzel, C. Daniels, A. Deutschländer, U. Dillmann, W. Eisner, *et al.*, "A randomized trial of deep-brain stimulation for parkinson's disease," *New England Journal of Medicine*, vol. 355, no. 9, pp. 896–908, 2006.

[6] K. A. Follett, F. M. Weaver, M. Stern, K. Hur, C. L. Harris, P. Luo, W. J. Marks Jr, J. Rothlind, O. Sagher, C. Moy, *et al.*, "Pallidal versus subthalamic deep-brain stimulation for parkinson's disease," *New England Journal of Medicine*, vol. 362, no. 22, pp. 2077–2091, 2010.

[7] M. S. Okun, "Deep-brain stimulation for parkinson's disease," *New England Journal of Medicine*, vol. 367, no. 16, pp. 1529–1538, 2012.

[8] J. Pineau, A. Guez, R. Vincent, G. Panuccio, and M. Avoli, "Treating epilepsy via adaptive neurostimulation: a reinforcement learning approach," *International journal of neural systems*, vol. 19, no. 04, pp. 227–240, 2009.

[9] M. Beudel and P. Brown, "Adaptive deep brain stimulation in parkinson's disease," *Parkinsonism & related disorders*, vol. 22, pp. S123–S126, 2016.

[10] M. Arlotti, L. Rossi, M. Rosa, S. Marceglia, and A. Priori, "An external portable device for adaptive deep brain stimulation (adbs) clinical research in advanced parkinson's disease," *Medical engineering & physics*, vol. 38, no. 5, pp. 498–505, 2016.

[11] M. Arlotti, M. Rosa, S. Marceglia, S. Barbieri, and A. Priori, "The adaptive deep brain stimulation challenge," *Parkinsonism & related disorders*, vol. 28, pp. 12–17, 2016.

[12] S. Little, E. Tripoliti, M. Beudel, A. Pogosyan, H. Cagnan, D. Herz, S. Bestmann, T. Aziz, B. Cheeran, L. Zrinzo, *et al.*, "Adaptive deep brain stimulation for parkinson's disease demonstrates reduced speech side effects compared to conventional stimulation in the acute setting," *J Neurol Neurosurg Psychiatry*, vol. 87, no. 12, pp. 1388–1389, 2016.

[13] S. Little, A. Pogosyan, S. Neal, B. Zavala, L. Zrinzo, M. Hariz, T. Foltynie, P. Limousin, K. Ashkan, J. FitzGerald, *et al.*, "Adaptive deep brain stimulation in advanced parkinson disease," *Annals of neurology*, vol. 74, no. 3, pp. 449–457, 2013.

[14] J. G. Habets, M. Heijmans, M. L. Kuijf, M. L. Janssen, Y. Temel, and P. L. Kubben, "An update on adaptive deep brain stimulation in parkinson's disease," *Movement Disorders*, vol. 33, no. 12, pp. 1834–1843, 2018.

[15] A. Guez, R. D. Vincent, M. Avoli, and J. Pineau, "Adaptive treatment of epilepsy via batch-mode reinforcement learning." in *AAAI*, 2008, pp. 1671–1678.

[16] V. Nagaraj, A. Lamperski, and T. I. Netoff, "Seizure control in a computational model using a reinforcement learning stimulation paradigm," *International journal of neural systems*, vol. 27, no. 07, p. 1750012, 2017.

[17] B. Colder, "The basal ganglia select the expected sensory input used for predictive coding," *Frontiers in computational neuroscience*, vol. 9, p. 119, 2015.

[18] D. T. Brocker, B. D. Swan, R. Q. So, D. A. Turner, R. E. Gross, and W. M. Grill, "Optimized temporal pattern of brain stimulation designed by computational evolution," *Science Translational Medicine*, vol. 9, no. 371, 2017.

[19] K. Kumaravelu, D. T. Brocker, and W. M. Grill, "A biophysical model of the cortex-basal ganglia-thalamus network in the 6-ohda lesioned rat model of parkinson's disease," *Journal of computational neuroscience*, vol. 40, no. 2, pp. 207–229, 2016.

[20] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," 2014.

[21] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[22] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[23] Y. Wu, E. Mansimov, R. B. Grosse, S. Liao, and J. Ba, "Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation," in *Advances in neural information processing systems*, 2017, pp. 5279–5288.

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

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, p. 529, 2015.

[26] N. Casas, "Deep deterministic policy gradient for urban traffic light control," *arXiv preprint arXiv:1703.09035*, 2017.

[27] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 3389–3396.

[28] I. Popov, N. Heess, T. Lillicrap, R. Hafner, G. Barth-Maron, M. Vecerik, T. Lampe, Y. Tassa, T. Erez, and M. Riedmiller, "Data-efficient deep reinforcement learning for dexterous manipulation," *arXiv preprint arXiv:1704.03073*, 2017.

[29] J. Redding, A. Geramifard, and J. How, "Actor-critic policy learning in cooperative planning," in *2010 AAAI Spring Symposium Series*, 2010.

[30] R. Q. So, A. R. Kent, and W. M. Grill, "Relative contributions of local cell and passing fiber activation and silencing to changes in thalamic fidelity during deep brain stimulation and lesioning: a computational modeling study," *Journal of computational neuroscience*, vol. 32, no. 3, pp. 499–519, 2012.

[31] I. Jovanov, M. Naumann, K. Kumaravelu, W. M. Grill, and M. Pajic, "Platform for model-based design and testing for deep brain stimulation," in *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems*. IEEE Press, 2018, pp. 263–274.

[32] I. R. Cassar, N. D. Titus, and W. M. Grill, "An improved genetic algorithm for designing optimal temporal patterns of neural stimulation," *Journal of Neural Engineering*, vol. 14, no. 6, p. 066013, nov 2017.

[33] M. G. Lagoudakis and R. Parr, "Least-squares policy iteration," *Journal of machine learning research*, vol. 4, no. Dec, pp. 1107–1149, 2003.

[34] E. Parisotto, J. L. Ba, and R. Salakhutdinov, "Actor-mimic: Deep multitask and transfer reinforcement learning," *arXiv preprint arXiv:1511.06342*, 2015.

[35] 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.

[36] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[37] Y. Zheng, Q. Liu, E. Chen, Y. Ge, and J. L. Zhao, "Exploiting multi-channels deep convolutional neural networks for multivariate time series classification," *Frontiers of Computer Science*, vol. 10, no. 1, pp. 96–112, 2016.

[38] J. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep convolutional neural networks on multichannel time series for human activity recognition," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[39] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[40] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.