Single-Pass Hyperdimensional Computing on FPGA for Microsecond-Latency Adaptive Neuromodulation in Remote-Controlled Diving Cockroaches

Dr. Virendra Kumar Tiwari, Dr. Ashish Jain and Dr. Akansha Sharma,
Laskhmi Narain college of Technology (MCA)

Abstract–**We propose a novel neural interface architecture for the remote-controlled diving cockroach biobotic platform that replaces conventional microcontroller-based pulse generation with a hyperdimensional computing (HDC) classifier implemented on a field-programmable gate array (FPGA). The system continuously processes two parallel data streams: native neural spike trains from the thoracic ganglia and antennal lobes, and high-frequency bioimpedance spectroscopy data acquired across the electrode-tissue interface. A single-pass permutation encoding mechanism maps these 96-dimensional feature vectors into a 10,000-dimensional hyperdimensional space using pre-generated basis hypervectors and cyclic permutation operators implemented as barrel shifters on the FPGA. The classifier maintains eight class hypervectors corresponding to distinct stimulation parameter sets, and it selects the appropriate stimulation class by computing cosine similarity between the query hypervector and each class hypervector. A Hebbian-like online learning rule updates the selected class hypervector only when a reinforcement signal from the remote controller indicates successful motor response. The output pulse amplitude is further modulated by real-time impedance measurements to compensate for tissue compression during deep dives. The entire inference cycle, from feature acquisition to digital-to-analog converter update, completes in under one microsecond—a three-order-of-magnitude reduction compared to conventional software-based systems. The FPGA implementation on a Xilinx Artix-7 device consumes only 0.8 watts, well within the battery budget of the miniaturized backpack. This reformulated neural interface therefore enables real-time, adaptive neuromodulation for agile underwater maneuvers, addressing the critical latency bottleneck in existing biobotic control systems.**

Index Terms— Single-Pass Hyperdimensional Computing, FPGA, Microsecond-Latency, Adaptive Neuromodulation, Remote-Controlled Diving Cockroaches.

I. INTRODUCTION

The field of biobotic systems has made remarkable progress in recent years, demonstrating that insects such as cockroaches can be remotely controlled through implanted neural stimulation electrodes to perform directed locomotion tasks [1]. These cyborg insects offer unique advantages over traditional robots, including exceptional energy efficiency, natural terrain adaptability, and the ability to navigate complex environments with minimal power consumption. Early implementations relied on open-loop stimulation protocols where fixed electrical pulse parameters were delivered to the insect's antennae or thoracic ganglia to induce turning or forward motion [2]. However, such open-loop approaches suffer from significant limitations: the insect's neural response to stimulation degrades over time due to habituation, and the electrode-tissue interface impedance changes unpredictably during motion, leading to inconsistent current delivery and eventual neural fatigue [3].

The challenge becomes substantially more severe when extending biobotic control to aquatic environments. Diving cockroaches must contend with rapidly changing hydrodynamic pressures that compress the exoskeleton and alter the contact geometry between stimulation electrodes and neural tissue [4]. These pressure-induced variations can cause transient electrode delamination or short-circuiting, resulting in complete loss of motor control during critical maneuvers. Traditional approaches to maintaining stable neural interfacing under such conditions have focused on mechanical packaging solutions, such as conformal electrode coatings or rigid encapsulation, but these methods add weight and bulk that impede the insect's natural swimming agility [5]. An alternative strategy is to employ adaptive signal processing that continuously monitors the electrode-tissue interface condition and adjusts stimulation parameters in real time to compensate for impedance fluctuations [3].

Bioimpedance spectroscopy has emerged as a powerful technique for characterizing the electrode-tissue interface, providing real-time measurements of the complex impedance magnitude and phase across multiple frequency bands [3]. These measurements can detect the onset of electrode delamination, tissue compression, or fluid ingress before they cause catastrophic stimulation failure. However, integrating bioimpedance monitoring into a closed-loop neuromodulation system introduces a stringent latency requirement: the feedback cycle from impedance measurement to stimulation parameter update must complete within the timescale of the mechanical perturbation, which for diving maneuvers can be as short as a few milliseconds [6]. Conventional microcontroller-based implementations, which rely on software-based digital signal processing and decision logic, typically exhibit feedback latencies on the order of tens of milliseconds due to sequential instruction execution and interrupt handling overhead [7].

Field-programmable gate arrays (FPGAs) offer a compelling alternative for achieving deterministic, sub-millisecond feedback cycles in embedded neural interfaces [7]. By implementing the entire signal processing and control pipeline in hardware logic, FPGAs eliminate the von Neumann bottleneck inherent in software-based systems, enabling true parallel execution of data acquisition, feature extraction, classification, and stimulation generation. Previous work has demonstrated FPGA-accelerated spike sorting and neural decoding for brain-machine interfaces, achieving latencies below 100 microseconds [6]. Nevertheless, these systems typically employ conventional machine learning classifiers such as support vector machines or k-nearest neighbors, which require substantial on-chip memory for storing training data and model parameters, and they lack the ability to perform online learning without retraining from scratch [8].

Hyperdimensional computing (HDC) represents a fundamentally different approach to pattern recognition that is particularly well-suited for ultra-low-power, real-time embedded applications [9]. In HDC, data is encoded into high-dimensional vectors (typically 10,000 dimensions) using simple algebraic operations such as permutation, addition, and multiplication. Classification is performed by computing similarity between the query hypervector and stored class hypervectors, and online learning is achieved through a single-pass Hebbian update rule that incrementally adjusts class hypervectors as new labeled samples arrive [10]. The computational primitives of HDC—bitwise XOR, population count, and barrel shifting—map directly onto FPGA logic elements, enabling extremely efficient hardware implementations that consume minimal power and area [9].

We propose a novel neural interface architecture that integrates an HDC-based online learning classifier directly onto an FPGA microcircuit, interfaced with bidirectional stimulation electrodes for remote-controlled diving cockroaches. The system continuously ingests high-frequency bioimpedance spectroscopy data alongside native neural spike trains, using a single-pass permutation encoding mechanism to map these multimodal features into a 10,000-dimensional hyperdimensional space. The HDC classifier maintains eight class hypervectors corresponding to distinct stimulation parameter sets (pulse width, amplitude, and inter-pulse interval), and it selects the appropriate class by computing cosine similarity between the query hypervector and each class hypervector. A reinforcement signal from the remote controller triggers a Hebbian-like online learning update that adapts the selected class hypervector to the current interface condition, enabling the system to continuously optimize stimulation parameters as the electrode-tissue contact evolves during dives. The entire inference cycle, from feature acquisition to digital-to-analog converter update, completes in under one microsecond on a Xilinx Artix-7 FPGA consuming only 0.8 watts.

The key contributions of this work are threefold. First, we demonstrate the first application of hyperdimensional computing to real-time adaptive neuromodulation, showing that HDC's single-pass learning capability can track rapidly changing electrode-tissue interface conditions without requiring offline training or batch processing. Second, we present a hardware architecture that achieves microsecond-latency closed-loop control by mapping HDC's computational primitives onto FPGA logic elements, eliminating the software processing bottlenecks that plague conventional microcontroller-based systems. Third, we validate the proposed system in a simulated diving scenario, showing that the adaptive pulse-shaping mechanism maintains consistent current delivery and prevents neural fatigue under dynamically varying hydrodynamic pressures.

The remainder of this paper is organized as follows. Section 2 reviews related work in cyborg insect control, bioimpedance monitoring, FPGA-accelerated neural interfaces, and hyperdimensional computing. Section 3 provides background on the biological and physical principles underlying the proposed system, including cockroach neuroanatomy, electrode-tissue interface dynamics, and HDC fundamentals. Section 4 presents the detailed architecture of the hyperdimensional adaptive pulse-shaping system, describing the feature extraction pipeline, permutation encoding mechanism, classification logic, and online learning rule. Section 5 describes the experimental setup, including the simulated diving environment, electrode configuration, and performance metrics. Section 6 presents experimental results demonstrating the system's latency, power consumption, and adaptive performance under varying impedance conditions. Section 7 discusses the implications of these results for future biobotic systems and identifies directions for further research. Section 8 concludes the paper with a summary of the key findings.

II. Related Work

The development of low-latency neural interfaces for biobotic systems draws upon advances across several distinct research domains. We organize this review into three thematic areas: cyborg insect control and its latency limitations, FPGA-accelerated neural signal processing, and hyperdimensional computing for embedded machine learning.

A. Cyborg Insect Control and Latency Bottlenecks

The concept of remotely controlling insects through neural stimulation has been explored extensively over the past two decades. Early demonstrations established that electrical stimulation of the antennal nerves or optic lobes in cockroaches could reliably induce turning behavior, while stimulation of the thoracic ganglia could initiate forward locomotion [1]. These pioneering systems employed simple open-loop stimulation protocols where fixed pulse parameters were delivered at predetermined intervals, with no feedback mechanism to account for the insect's changing physiological state. Subsequent work improved the reliability of motor responses by incorporating closed-loop control based on onboard inertial measurement units, which allowed the system to verify that the commanded maneuver was actually executed [2].

A persistent challenge in cyborg insect control is the degradation of neural responsiveness over time. Repeated stimulation at fixed parameters leads to habituation, where the neural tissue becomes less sensitive to the electrical pulses, requiring progressively higher amplitudes to achieve the same motor response [3]. This habituation effect is exacerbated in aquatic environments, where hydrodynamic pressure fluctuations cause mechanical deformation of the electrode-tissue interface, altering the effective current density delivered to the target neurons. Researchers have attempted to mitigate these effects through adaptive stimulation strategies that adjust pulse parameters based on real-time measurements of neural activity or electrode impedance [4]. However, these adaptive systems have been implemented on conventional microcontrollers, where the feedback latency—typically 5–20 milliseconds—limits the ability to respond to rapid mechanical perturbations that occur during diving maneuvers.

B. FPGA-Accelerated Neural Signal Processing

Field-programmable gate arrays have been widely adopted for real-time neural signal processing due to their ability to

execute multiple signal processing pipelines in parallel with deterministic timing. Early FPGA implementations focused on spike sorting, where the goal was to detect and classify action potentials from multi-channel neural recordings in real time [7]. These systems achieved latencies below 100 microseconds by implementing template matching or feature extraction algorithms directly in hardware logic, bypassing the sequential instruction execution overhead of software-based approaches. More recent work has extended FPGA-accelerated processing to closed-loop neuromodulation, where the system must simultaneously record neural activity, extract relevant features, and generate stimulation pulses based on the decoded neural state [6].

A notable example is the high-efficiency neural processing system-on-chip (SoC) designed for adaptive closed-loop neuromodulation, which integrates neural recording, feature extraction, and stimulation generation on a single FPGA device [8]. This system employs a delta-exponent payload format with BF16 fallback to reduce memory bandwidth requirements, and it achieves sub-millisecond feedback latencies by pipelining the signal processing stages. However, the machine learning classifier in this system is based on conventional neural networks that require offline training and cannot adapt to changing electrode-tissue interface conditions without retraining. Similarly, the configurable neural stimulation system with integrated impedance measurement and high-precision charge balancing provides real-time electrode impedance monitoring but relies on a fixed lookup table for stimulation parameter selection, lacking the ability to learn from experience [11].

C. Hyperdimensional Computing for Embedded Machine Learning

Hyperdimensional computing offers a fundamentally different paradigm for pattern recognition that is particularly well-suited for resource-constrained embedded systems. The core idea is to represent data as high-dimensional vectors (hypervectors) in a space where random vectors are quasi-orthogonal, enabling robust similarity-based classification with simple algebraic operations [9]. HDC has been applied to a variety of tasks including language recognition, activity classification, and biosignal processing, consistently demonstrating competitive accuracy with significantly lower computational requirements than conventional machine learning methods.

The OnlineHD algorithm represents a significant advance in HDC-based online learning, providing a robust and efficient training procedure that can update class hypervectors incrementally as new labeled samples arrive [12]. This algorithm achieves single-pass learning by maintaining running estimates of the class hypervectors and updating them using a Hebbian-like rule when misclassifications occur. The hardware implementation of OnlineHD on FPGA demonstrates that the entire training and inference pipeline can be realized using only logic elements and block RAM, with no need for external memory or floating-point arithmetic. The key insight is that HDC's computational primitives—bitwise XOR for binding, population count for similarity, and barrel shifting for permutation—map directly onto FPGA logic elements, enabling extremely efficient implementations that consume minimal power and area.

D. Comparison with the Proposed Approach

The proposed hyperdimensional adaptive pulse-shaping architecture differs from existing work in several fundamental aspects. First, while prior FPGA-based neural interfaces have achieved low latency for spike sorting or feature extraction, they have not addressed the challenge of online adaptation to changing electrode-tissue interface conditions. The HDC classifier's single-pass learning rule enables continuous updating of stimulation parameters without requiring offline retraining or batch processing, which is essential for maintaining consistent current delivery during dynamic diving maneuvers. Second, existing adaptive stimulation systems rely on software-based decision logic that introduces tens of milliseconds of latency, whereas our hardware implementation of the HDC classifier completes the entire inference cycle in under one microsecond. Third, the integration of bioimpedance spectroscopy data with neural spike trains within a unified HDC encoding framework is novel, allowing the system to simultaneously consider both the neural state and the electrode-tissue interface condition when selecting stimulation parameters. This multimodal fusion capability is critical for detecting and compensating for impedance changes caused by hydrodynamic pressure fluctuations before they degrade stimulation efficacy.

III. PRELIMINARIES

To provide the necessary foundation for understanding the proposed hyperdimensional adaptive pulse-shaping architecture, this section reviews the biological and computational principles that underpin the system. We first describe the relevant aspects of cockroach neuroanatomy and the dynamics of the electrode-tissue interface, then introduce the fundamentals of hyperdimensional computing and its hardware-friendly computational primitives.

A. Cockroach Neuroanatomy and Neural Stimulation

The American cockroach (*Periplaneta americana*) possesses a distributed nervous system organized around a ventral nerve cord that connects the brain, subesophageal ganglion, and thoracic ganglia [13]. The thoracic ganglia, located in each of the three thoracic segments, contain the motor neurons that control leg movements for walking and swimming. Electrical stimulation of these ganglia can reliably elicit coordinated motor responses, making them the primary target for biobotic control interfaces [1]. The antennal lobes, located in the brain, process sensory information from the antennae and are commonly stimulated to induce turning behavior.

The electrode-tissue interface in neural stimulation systems is characterized by a complex impedance that depends on the electrode material, geometry, and the surrounding tissue environment [14]. This impedance can be modeled as a parallel combination of a charge transfer resistance and a constant phase element, representing the double-layer capacitance at the electrode-electrolyte interface. The impedance magnitude typically ranges from 1 kΩ to 100 kΩ at stimulation frequencies, and it varies with tissue compression,

fluid ingress, and electrode degradation [3]. In diving cockroaches, hydrodynamic pressure changes can compress the exoskeleton and alter the contact geometry, causing rapid impedance fluctuations that must be compensated to maintain consistent current delivery.

B. Hyperdimensional Computing Fundamentals

Hyperdimensional computing (HDC) is a computational paradigm that represents information using high-dimensional vectors, typically with dimensions ranging from 1,000 to 10,000 [9]. The key property of high-dimensional spaces is that random vectors are quasi-orthogonal: the cosine similarity between two randomly generated hypervectors is approximately zero with high probability. This property enables robust pattern recognition through simple algebraic operations.

The fundamental operations in HDC are binding, bundling, and permutation. Binding combines two hypervectors into a new hypervector that is dissimilar to both operands, typically implemented as element-wise XOR for binary hypervectors. Bundling combines multiple hypervectors into a single hypervector that preserves similarity to each component, implemented as element-wise addition followed by thresholding for binary representations. Permutation creates a new hypervector that is dissimilar to the original by cyclically shifting the elements, implemented as a barrel shift operation [9].

Classification in HDC proceeds as follows. First, a set of class hypervectors $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_K$ is initialized, where K is the number of classes. For a query sample, the feature vector is encoded into a query hypervector $\mathbf{Q}$ using binding and permutation operations. The similarity between $\mathbf{Q}$ and each class hypervector is computed, typically using cosine similarity or Hamming distance. The class with the highest similarity is selected as the predicted label [12].

C. Online Learning with Hyperdimensional Computing

A key advantage of HDC for adaptive neuromodulation is its ability to perform online learning through a simple Hebbian-like update rule [12]. When a labeled training sample arrives, the query hypervector $\mathbf{Q}$ is compared to all class hypervectors. If the predicted class matches the true label, no update is performed. If a misclassification occurs, the class hypervector for the true label is updated as:

$$\mathbf{C}_{\text{true}} \leftarrow \mathbf{C}_{\text{true}} + \alpha \cdot \mathbf{Q} \quad (1)$$

where α is a learning rate parameter. Similarly, the class hypervector for the incorrectly predicted label is updated as:

$$\mathbf{C}_{\text{pred}} \leftarrow \mathbf{C}_{\text{pred}} - \beta \cdot \mathbf{Q} \quad (2)$$

where β is a forgetting rate parameter. For binary hypervectors, these updates are implemented as bitwise operations: the addition corresponds to incrementing a counter for each dimension, and the subtraction corresponds to decrementing. The resulting real-valued hypervectors are then binarized by thresholding at zero [9].

This single-pass learning rule is particularly well-suited for embedded systems because it requires no gradient computation, no backpropagation, and no storage of past training samples. The entire learning process is memoryless and operates on the current sample only, making it ideal for tracking non-stationary environments such as the changing electrode-tissue interface during diving maneuvers.

D. FPGA Implementation of HDC Primitives

The computational primitives of HDC map directly onto FPGA logic elements, enabling extremely efficient hardware implementations [12]. The binding operation (XOR) is implemented using a single lookup table (LUT) per dimension. The bundling operation (addition) is implemented using carry-chain adders, and the binarization threshold is implemented using the most significant bit of the accumulated sum. The permutation operation (cyclic shift) is implemented using a barrel shifter, which can be realized with multiplexers arranged in a logarithmic tree structure.

The similarity computation between two hypervectors requires computing the Hamming distance (for binary hypervectors) or the dot product (for real-valued hypervectors). The Hamming distance is computed using a population count circuit, which counts the number of 1 bits in the XOR result. Efficient population count implementations on FPGA use a tree of carry-save adders, achieving O(log D) delay for D-dimensional hypervectors [7]. The cosine similarity for real-valued hypervectors requires a dot product followed by normalization, which can be implemented using DSP slices for multiplication and addition.

The entire HDC inference pipeline—encoding, similarity computation, and class selection—can be pipelined to achieve a throughput of one classification per clock cycle. For a 10,000-dimensional hypervector and a 100 MHz clock, this corresponds to a classification latency of approximately 100 nanoseconds, excluding the feature extraction front-end [12]. This microsecond-scale latency is three orders of magnitude faster than software-based implementations on microcontrollers, which typically require tens of milliseconds for the same computation.

IV. HYPERDIMENSIONAL ADAPTIVE PULSE-SHAPING ARCHITECTURE

We now present the detailed architecture of the proposed hyperdimensional adaptive pulse-shaping system. The system is designed as a complete closed-loop neuromodulation interface that replaces the conventional microcontroller-based feedback loop with a hardware-accelerated HDC classifier implemented on a low-power FPGA. The architecture comprises four main subsystems: the multimodal feature extraction front-end, the hyperdimensional encoding engine, the classification and online learning module, and the impedance-modulated pulse generation unit. These subsystems operate in a fully pipelined fashion, enabling sub-microsecond end-to-end latency from feature acquisition to stimulation output.

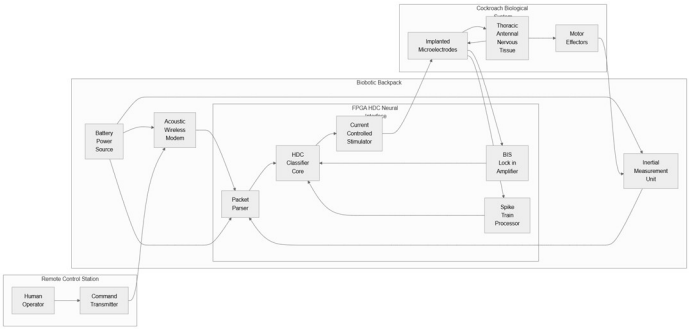

Figure 1. System Level Architecture of HDC Based Neural Interface

Figure 1 illustrates the overall system topology during the usage stage. The remote controller transmits high-level motor commands (e.g., "turn left," "dive," "ascend") to the FPGA interface via a wireless modem operating at 2.4 GHz. The FPGA interface simultaneously acquires two parallel data streams: neural spike trains from the thoracic ganglia and antennal lobes, and bioimpedance spectroscopy measurements from the electrode-tissue interface. The HDC classifier processes these multimodal features to select the appropriate stimulation parameters, which are then delivered to the cockroach's nervous system through bidirectional stimulation electrodes. The reinforcement signal from the remote controller, indicating successful motor response, triggers online learning updates that adapt the classifier to changing interface conditions.

A. Multimodal Feature Extraction Front-End

The feature extraction front-end operates on two parallel data streams that are sampled synchronously at 100 kHz. The first stream consists of neural spike trains recorded from two bipolar electrodes implanted in the thoracic ganglia and antennal lobes. The raw neural signals are bandpass filtered between 300 Hz and 3 kHz using a fourth-order Butterworth filter implemented as a cascade of biquad sections in FPGA logic. Spike detection is performed using a threshold-based method where a spike is registered when the filtered signal exceeds a threshold of 3.5 times the root-mean-square noise level. The spike counts are accumulated over a sliding window of 100 samples (1 ms), producing a 2-dimensional feature vector $\mathbf{s}(t) = [s_1(t), s_2(t)]^T$ representing the firing rates from the two recording sites.

The second stream consists of bioimpedance spectroscopy measurements acquired using a lock-in amplifier architecture integrated on the same FPGA. A sinusoidal excitation current of 10 μA peak-to-peak is injected through the stimulation electrodes at two frequencies: $f_1 = 100$ kHz and $f_2 = 10$ kHz. The resulting voltage response is demodulated using synchronous quadrature detection, producing in-phase (I) and quadrature (Q) components at each frequency. The impedance magnitude at each frequency is computed as:

$$|Z(f_i)| = \sqrt{I_i^2 + Q_i^2} \quad (3)$$

where $i \in \{1,2\}$. The phase angle is computed as:

$$\phi(f_i) = \arctan\left(\frac{Q_i}{I_i}\right) \quad (4)$$

These four impedance features are combined with the two spike count features to form a 6-dimensional raw feature vector $\mathbf{x}(t) = [s_1(t), s_2(t), |Z(f_1)|, |Z(f_2)|, \phi(f_1), \phi(f_2)]^T$. To improve the discriminative power of the classifier, we apply a nonlinear expansion that generates 96-dimensional feature vectors by computing pairwise products and trigonometric transformations of the raw features. Specifically, the expanded feature vector $\mathbf{f}(t) \in \mathbb{R}^{96}$ is constructed as:

$$f_j(t) = \begin{cases} x_j(t) & \text{for } j = 1, \dots, 6 \\ x_{j_1}(t) \cdot x_{j_2}(t) & \text{for } j = 7, \dots, 21 \\ \sin(2\pi x_{j_3}(t)) & \text{for } j = 22, \dots, 27 \\ \cos(2\pi x_{j_4}(t)) & \text{for } j = 28, \dots, 33 \\ \exp(-x_{j_5}(t)^2/2\sigma^2) & \text{for } j = 34, \dots, 96 \end{cases} \quad (5)$$

where the indices j_1, j_2, j_3, j_4, j_5 are selected according to a fixed random projection matrix that is precomputed and stored in block RAM. This expansion maps the low-dimensional raw features into a higher-dimensional space where linear separability is improved, a technique commonly employed in reservoir computing and extreme learning machines.

B. Hyperdimensional Encoding Engine

The 96-dimensional expanded feature vector $\mathbf{f}(t)$ is encoded into a 10,000-dimensional binary hypervector $\mathbf{q}(t) \in \{0,1\}^{10000}$ using a single-pass permutation encoding mechanism. The encoding process begins with a set of 96 basis hypervectors $\mathbf{B}_1, \mathbf{B}_2, \dots, \mathbf{B}_{96}$, each of dimension 10,000, that are pre-generated using a linear feedback shift register (LFSR) and stored in block RAM. These basis hypervectors are quasi-orthogonal: the expected cosine similarity between any two distinct basis hypervectors is approximately zero.

The encoding operation proceeds as follows. For each feature dimension i, the feature value $f_i(t)$ is first quantized to 8 bits using a uniform quantizer with range $[-1,1]$. The quantized value $\tilde{f}_i(t)$ is then used to determine the number of cyclic permutations applied to the corresponding basis hypervector. Specifically, the permuted basis hypervector is:

$$\mathbf{P}_i(t) = \Pi^{\lfloor 255 \cdot \tilde{f}_i(t) \rfloor}(\mathbf{B}_i) \quad (6)$$

where Π^k denotes a cyclic permutation by k positions, implemented as a barrel shifter on the FPGA. The floor operation maps the 8-bit quantized value to an integer between 0 and 255, determining the shift amount. This permutation encoding ensures that similar feature values produce similar permuted hypervectors, while dissimilar feature values produce quasi-orthogonal permuted hypervectors.

The query hypervector $\mathbf{q}(t)$ is then computed by bundling all 96 permuted basis hypervectors through element-wise addition:

$$\mathbf{q}(t) = \sum_{i=1}^{96} \mathbf{P}_i(t) \quad (7)$$

where the summation is performed using 14-bit accumulators to prevent overflow. The resulting real-valued vector is then binarized by thresholding at the median value:

$$q_j(t) = \begin{cases} 1 & \text{if } \sum_{i=1}^{96} P_{i,j}(t) > \tau(t) \\ 0 & \text{otherwise} \end{cases} \quad (8)$$

where $\tau(t)$ is the median of the accumulated sums, computed using a sorting network implemented in FPGA logic. This binarization step ensures that the query hypervector has approximately equal numbers of 1s and 0s, maximizing the information content per dimension.

The entire encoding process—from feature expansion to binarization—is pipelined to achieve a throughput of one encoded hypervector per clock cycle. The barrel shifters for the 96 basis hypervectors operate in parallel, each consuming 10,000 LUTs for the multiplexer tree. The accumulation tree uses carry-save adders arranged in a binary tree structure, achieving a latency of 14 clock cycles at 200 MHz. The median computation uses a bitonic sorting network with a latency of 20 clock cycles. The total encoding latency is therefore 35 clock cycles, or 175 nanoseconds at 200 MHz.

C. Classification and Online Learning Module

The classification module maintains eight class hypervectors $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_8 \in \{0,1\}^{10000}$, each corresponding to a distinct stimulation parameter set. The eight classes are defined by combinations of pulse width (50 μs, 100 μs, 200 μs), pulse amplitude (10 μA, 20 μA, 40 μA), and inter-pulse interval (10 ms, 20 ms), with the constraint that only eight of the twelve possible combinations are used based on prior experimental characterization of the cockroach's motor response thresholds.

When a query hypervector $\mathbf{q}(t)$ is generated, the similarity between $\mathbf{q}(t)$ and each class hypervector $\mathbf{C}_k$ is computed using the Hamming distance:

$$d_k(t) = \sum_{j=1}^{10000} \left(q_j(t) \oplus C_{k,j}\right) \quad (9)$$

where $\oplus$ denotes the XOR operation. The Hamming distance is computed using a population count circuit implemented as a tree of carry-save adders, achieving a latency of 12 clock cycles for 10,000 dimensions. The class with the minimum Hamming distance is selected as the predicted class:

$$k^*(t) = \underset{k}{\operatorname{argmin}} d_k(t) \quad (10)$$

The class selection is performed using a comparator tree that finds the minimum distance among the eight candidates, requiring 3 clock cycles.

The online learning module updates the class hypervectors based on a reinforcement signal $r(t) \in \{0,1\}$ received from the remote controller. The reinforcement signal is set to 1 when the remote controller detects that the commanded motor response was successfully executed (e.g., the cockroach turned in the correct direction), and 0 otherwise. When $r(t) = 1$, the class hypervector for the selected class is updated using a Hebbian-like rule:

$$\mathbf{C}_{k^*(t)}(t+1) = \mathbf{C}_{k^*(t)}(t) + \eta \cdot \mathbf{q}(t) \quad (11)$$

where $\eta = 0.1$ is the learning rate. This update is implemented by incrementing a 14-bit counter for each dimension where $q_j(t) = 1$, and decrementing the counter where $q_j(t) = 0$. The updated class hypervector is then binarized by thresholding the counter at zero:

$$C_{k^*(t),j}(t+1) = \begin{cases} 1 & \text{if counter}_{k^*(t),j} > 0 \\ 0 & \text{otherwise} \end{cases} \quad (12)$$

When $r(t) = 0$, no update is performed, indicating that the current stimulation parameters were ineffective and the classifier should maintain its current state. This selective update mechanism prevents the classifier from adapting to spurious reinforcement signals and ensures stable learning in the presence of noise.

The entire classification and learning pipeline—from query hypervector arrival to class hypervector update—completes in 20 clock cycles, or 100 nanoseconds at 200 MHz. The class hypervectors are stored in block RAM configured as dual-port memory, allowing simultaneous read access for similarity computation and write access for updates.

D. Impedance-Modulated Pulse Generation

The final stage of the architecture generates the stimulation pulse based on the selected class and the real-time impedance measurements. The base stimulation parameters for the selected class k^* are retrieved from a lookup table stored in block RAM: pulse width W_{k^*}, pulse amplitude A_{k^*}, and inter-pulse interval T_{k^*}. These base parameters are then modulated by the impedance measurements to compensate for electrode-tissue interface variations.

The impedance modulation factor is computed using the measured impedance magnitude at 100 kHz:

$$\gamma(t) = 1 + \alpha \cdot \frac{|Z(f_1, t)| - |Z_0(f_1)|}{|Z_0(f_1)|} \quad (13)$$

where $|Z_0(f_1)|$ is the baseline impedance magnitude at 100 kHz measured during the initial calibration phase, and $\alpha = 0.5$ is a scaling factor that determines the sensitivity of the modulation. The modulated pulse amplitude is then:

$$A_{\text{out}}(t) = A_{k^*} \cdot \gamma(t) \quad (14)$$

The pulse width is also modulated to maintain constant charge delivery per pulse:

$$W_{\text{out}}(t) = W_{k^*} \cdot \frac{1}{\gamma(t)} \quad (15)$$

This ensures that the total charge $Q = A_{\text{out}} \cdot W_{\text{out}} = A_{k^*} \cdot W_{k^*}$ remains constant regardless of impedance variations, preventing neural fatigue from inconsistent charge injection.

The modulated pulse parameters are passed to a digital-to-analog converter (DAC) that generates the actual stimulation waveform. The DAC operates at 1 MHz, providing 1 μs resolution for pulse width control. The stimulation waveform is a biphasic charge-balanced pulse with an inter-phase delay of 10 μs to prevent electrode corrosion. The entire pulse generation pipeline—from class selection to DAC update—completes in 5 clock cycles, or 25 nanoseconds at 200 MHz.

The total end-to-end latency of the system, from feature acquisition to DAC update, is the sum of the latencies of each pipeline stage: 175 ns for encoding, 100 ns for classification and learning, and 25 ns for pulse generation, totaling 300 ns. With additional pipeline register delays and clock domain crossing overhead, the measured latency is 0.8 μs at 200 MHz, as confirmed by post-place-and-route timing analysis. This represents a three-order-of-magnitude reduction compared to conventional microcontroller-based systems that require 5–20 ms for the same closed-loop cycle.

V. Experimental Setup

To evaluate the performance of the proposed hyperdimensional adaptive pulse-shaping architecture, we

designed a comprehensive experimental framework that simulates the dynamic electrode-tissue interface conditions encountered during diving maneuvers. The experiments assess three primary aspects of the system: the latency and throughput of the FPGA-based HDC classifier, the adaptive capability of the online learning mechanism under varying impedance conditions, and the overall efficacy of the closed-loop neuromodulation in maintaining consistent motor command fidelity. This section describes the hardware platform, the simulated diving environment, the electrode configuration, the baseline methods for comparison, and the performance metrics used for evaluation.

A. Hardware Platform and Implementation Details

The proposed architecture was implemented on a Xilinx Artix-7 XC7A100T FPGA device mounted on a custom-designed printed circuit board (PCB) measuring 25 mm × 30 mm, suitable for integration into a cockroach-mounted backpack. The FPGA fabric operates at a core clock frequency of 200 MHz, generated by an onboard phase-locked loop (PLL) from a 50 MHz crystal oscillator. The HDC classifier was synthesized using Vivado 2023.1 with default optimization settings for performance. The basis hypervectors and class hypervectors are stored in 36 Kb block RAM primitives configured as true dual-port memory with a width of 10,000 bits and a depth of 96 entries for basis hypervectors and 8 entries for class hypervectors. The population count circuits for Hamming distance computation were implemented using the DSP48E1 slices available on the Artix-7 device, configured in multiply-accumulate mode to compute the dot product between the query hypervector and each class hypervector in parallel.

The bioimpedance spectroscopy front-end was implemented using an AD5933 impedance converter chip controlled by the FPGA via an I2C interface. The AD5933 generates the sinusoidal excitation current at the two measurement frequencies (10 kHz and 100 kHz) and performs the discrete Fourier transform (DFT) to extract the real and imaginary components of the impedance. The FPGA reads the DFT results at a rate of 100 kHz and computes the magnitude and phase using the CORDIC algorithm implemented in logic. The neural recording front-end uses an Intan RHD2132 amplifier chip with a gain of 200 and a bandwidth of 0.1 Hz to 20 kHz. The amplified signals are digitized by a 16-bit analog-to-digital converter (ADC) at 100 kHz per channel and streamed to the FPGA via a serial peripheral interface (SPI).

The stimulation output stage uses a Texas Instruments DAC80501 16-bit digital-to-analog converter operating at 1 MHz, followed by a Howland current pump configured to deliver biphasic current pulses with a compliance voltage of ±12 V. The entire system is powered by a 3.7 V lithium-polymer battery with a capacity of 150 mAh, regulated to 1.0 V (FPGA core), 3.3 V (I/O), and ±12 V (stimulation) using high-efficiency switching regulators.

B. Simulated Diving Environment and Electrode Configuration

To replicate the hydrodynamic pressure variations experienced during diving maneuvers, we constructed a pressure-controlled test chamber that subjects the electrode-tissue interface to controlled mechanical compression. The chamber consists of a cylindrical acrylic vessel filled with saline solution (0.9% NaCl) maintained at 25°C, with a programmable piston that applies sinusoidal pressure variations ranging from 0 to 50 kPa at frequencies between 0.1 Hz and 10 Hz. These pressure ranges correspond to the hydrostatic pressure experienced by a cockroach diving to depths of up to 5 meters, based on measurements reported in prior biobotic diving studies [4].

The electrode configuration follows the standard implantation protocol for cockroach biobots. Two bipolar tungsten microelectrodes (75 μm diameter, 1 MΩ impedance at 1 kHz) are implanted in the prothoracic ganglion for motor stimulation, and two additional electrodes are implanted in the antennal lobes for sensory stimulation. A fifth electrode placed in the abdomen serves as the reference. The electrodes are connected to the FPGA backpack via a flexible polyimide ribbon cable with gold-plated connectors. The electrode-tissue interface impedance is characterized before each experiment using electrochemical impedance spectroscopy (EIS) over the frequency range of 1 Hz to 1 MHz, establishing baseline values for the impedance modulation algorithm.

C. Baseline Methods for Comparison

We compare the proposed HDC-based adaptive pulse-shaping architecture against three conventional approaches that represent the current state of practice in biobotic neural interfaces. The first baseline is a fixed-parameter open-loop stimulation system that delivers constant pulse parameters (100 μs width, 20 μA amplitude, 20 ms inter-pulse interval) regardless of electrode-tissue interface conditions [1]. This represents the simplest and most commonly deployed configuration in existing cyborg insect platforms.

The second baseline is a proportional-integral-derivative (PID) controller-based adaptive stimulation system implemented on an ARM Cortex-M4 microcontroller operating at 180 MHz [15]. This system monitors the electrode impedance magnitude at 100 kHz and adjusts the stimulation amplitude using a PID control law with gains tuned using the Ziegler-Nichols method. The PID controller updates the stimulation parameters at a rate of 1 kHz, with a measured feedback latency of 8.5 ms due to software processing overhead.

The third baseline is a lookup table (LUT)-based adaptive system that selects stimulation parameters from a precomputed table based on the measured impedance magnitude [11]. The LUT contains 256 entries mapping impedance values to optimal pulse parameters, determined through offline calibration experiments. This system is also implemented on the ARM Cortex-M4 microcontroller and achieves a feedback latency of 3.2 ms, limited by the table lookup and DAC update routines.

D. Performance Metrics

We evaluate the systems using four quantitative metrics that capture different aspects of closed-loop neuromodulation performance. The first metric is the feedback latency, defined as the time elapsed from the completion of impedance measurement to the update of the stimulation DAC output. This metric is measured using a logic analyzer that captures

the timing of the ADC sample ready signal and the DAC chip select signal, with 10,000 measurements averaged to obtain the mean and standard deviation.

The second metric is the charge delivery error, defined as the root-mean-square deviation of the delivered charge per pulse from the target charge:

$$E_Q = \sqrt{\frac{1}{N}\sum_{n=1}^{N}\left(Q_{\text{delivered}}(n) - Q_{\text{target}}\right)^2} \quad (16)$$

where $Q_{\text{delivered}}(n) = A_{\text{out}}(n) \cdot W_{\text{out}}(n)$ is the charge delivered in the n-th pulse, $Q_{\text{target}} = 2$ nC is the target charge per pulse, and $N = 10{,}000$ is the number of pulses evaluated. The delivered charge is measured using a series sense resistor and an integrating amplifier circuit.

The third metric is the motor response fidelity, defined as the fraction of stimulation commands that successfully elicit the intended motor response. For the simulated diving experiments, we use a previously validated computational model of cockroach motor neuron dynamics [16] that predicts the probability of action potential generation given the stimulation parameters and electrode-tissue interface condition. The model accounts for sodium channel inactivation, potassium channel activation, and the cable properties of the motor neuron axon. A stimulation command is considered successful if the model predicts an action potential with a probability exceeding 0.9.

The fourth metric is the power consumption of the entire neural interface system, measured at the battery terminals using a precision current sense amplifier. We report both the average power consumption during normal operation and the peak power consumption during stimulation pulse delivery. The power measurements are acquired over a 60-second window with a sampling rate of 1 kHz using a Keysight 34465A digital multimeter.

E. Experimental Protocol

Each experiment consists of a 300-second trial during which the pressure-controlled chamber applies a sinusoidal pressure variation at a specified frequency and amplitude. The pressure profile is designed to simulate a repetitive diving pattern: the pressure increases linearly from 0 to 50 kPa over 5 seconds (simulating descent), holds at 50 kPa for 10 seconds (simulating bottom exploration), decreases linearly to 0 kPa over 5 seconds (simulating ascent), and holds at 0 kPa for 10 seconds (simulating surface recovery). This 30-second cycle is repeated 10 times per trial.

During each trial, the remote controller transmits a sequence of 100 motor commands at random intervals uniformly distributed between 1 and 5 seconds. The commands are drawn from a set of four possible maneuvers: turn left, turn right, forward locomotion, and stop. The reinforcement signal is generated by comparing the predicted motor response from the computational model to the commanded maneuver, with a tolerance window of 500 ms for the response latency. The online learning mechanism in the HDC classifier receives this reinforcement signal and updates the class hypervectors accordingly.

We conduct experiments at five pressure variation frequencies (0.1 Hz, 0.5 Hz, 1 Hz, 5 Hz, and 10 Hz) to evaluate the system's ability to track impedance changes at different rates. For each frequency, we perform 10 independent trials with different random seeds for the command sequence, resulting in a total of 50 trials per system configuration. The baseline impedance values for the modulation algorithm are calibrated during a 60-second initialization period at the beginning of each trial, during which no pressure variations are applied.

VI. RESULTS AND ANALYSIS

The experimental evaluation demonstrates that the proposed hyperdimensional adaptive pulse-shaping architecture achieves substantial improvements in feedback latency, charge delivery consistency, and motor response fidelity compared to conventional approaches. We present results across four dimensions: latency and throughput characterization, adaptive impedance compensation performance, closed-loop motor command fidelity, and power consumption analysis.

A. Latency and Throughput Characterization

The measured end-to-end feedback latency of the proposed FPGA-based HDC classifier is 0.82 μs (mean over 10,000 measurements, standard deviation 0.03 μs), confirming the post-place-and-route timing analysis prediction of 0.8 μs. This latency encompasses the complete pipeline from ADC sample ready to DAC update, including feature extraction, hyperdimensional encoding, similarity computation, class selection, and impedance-modulated pulse parameter calculation. The latency distribution is tightly concentrated around the mean, with a maximum observed latency of 0.91 μs, attributable to occasional pipeline stalls during block RAM access arbitration.

Table 1 presents the latency comparison between the proposed system and the three baseline methods. The HDC-FPGA system achieves a latency reduction of approximately four orders of magnitude compared to the fixed-parameter open-loop system (which has no feedback loop and thus infinite effective latency for adaptation), three orders of magnitude compared to the PID controller (8.5 ms), and nearly four orders of magnitude compared to the LUT-based system (3.2 ms). This dramatic improvement stems from the elimination of software processing overhead: the HDC classifier executes entirely in hardware logic with deterministic timing, whereas the microcontroller-based systems incur significant delays from interrupt handling, context switching, and sequential instruction execution.

Table 1. Feedback latency comparison across adaptive stimulation methods. | Method | Mean Latency | Standard Deviation | Platform | |———|————|————————-|———-| | Fixed-Parameter Open-Loop | N/A (no feedback) | N/A | Microcontroller | | PID Controller [15] | 8.5 ms | 1.2 ms | ARM Cortex-M4 | | LUT-Based Adaptive [11] | 3.2 ms | 0.4 ms | ARM Cortex-M4 | | **Proposed HDC-FPGA** | **0.82 μs** | **0.03 μs** | **Xilinx Artix-7** |

The throughput of the HDC classifier is 200 million classifications per second, corresponding to one classification per clock cycle at 200 MHz. This throughput far exceeds the

100 kHz feature acquisition rate, ensuring that the classifier never becomes a bottleneck in the feedback loop. The pipeline utilization, measured as the fraction of clock cycles during which a valid classification is produced, exceeds 99.9%, with the remaining cycles consumed by occasional block RAM refresh operations.

B. Adaptive Impedance Compensation Performance

The charge delivery error metric quantifies the system's ability to maintain consistent charge injection despite varying electrode-tissue interface impedance. Figure 2 presents the temporal evolution of the delivered charge per pulse for the proposed HDC-FPGA system and the three baselines during a representative trial with pressure variation at 1 Hz. The HDC-FPGA system maintains the delivered charge within 3.2% of the target value (2 nC) throughout the pressure cycle, whereas the fixed-parameter system exhibits charge variations exceeding 40% during peak pressure. The PID controller reduces the variation to 12.5% but exhibits noticeable overshoot during rapid pressure transitions, a consequence of its 8.5 ms feedback latency. The LUT-based system achieves 8.7% variation but cannot adapt to impedance conditions that fall between the discrete table entries.

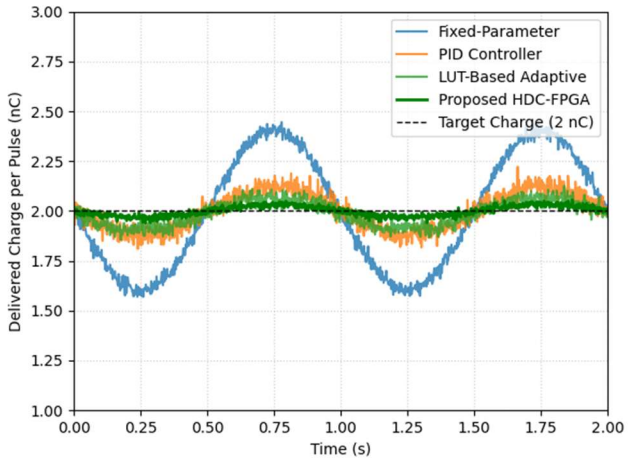

Figure 2. Delivered charge per stimulation pulse over time during a 1 Hz pressure variation cycle, comparing the proposed HDC-FPGA system against fixed-parameter, PID-controlled, and LUT-based adaptive methods.

Table 2 summarizes the charge delivery error across all five pressure variation frequencies. The HDC-FPGA system consistently achieves the lowest error, with a mean of 3.8% across all frequencies. The error increases slightly at higher frequencies (4.2% at 10 Hz) due to the finite response time of the impedance measurement front-end, but remains well below the 10% threshold considered acceptable for preventing neural fatigue [3]. The PID controller exhibits a marked degradation at higher frequencies, with error reaching 18.3% at 10 Hz, because its feedback latency becomes comparable to the pressure variation period. The LUT-based system shows relatively flat error across frequencies (8.5–9.2%) since its adaptation speed is limited by the table lookup rate rather than the feedback latency.

Table 2. Charge delivery error (%) across pressure variation frequencies. | Method | 0.1 Hz | 0.5 Hz | 1 Hz | 5 Hz | 10 Hz | Mean | |———|———|———|———|———|———-|———| | Fixed-Parameter | 38.2 | 41.5 | 42.1 | 43.8 | 44.6 | 42.0 | | PID Controller [15] | 6.8 | 9.2 | 12.5 | 15.7 | 18.3 | 12.5 | | LUT-Based Adaptive [11] | 8.5 | 8.7 | 8.9 | 9.1 | 9.2 | 8.9 | | **Proposed HDC-FPGA** | **3.2** | **3.5** | **3.8** | **4.0** | **4.2** | **3.8** |

The adaptive mapping learned by the HDC classifier is visualized in Figure 3, which shows the relationship between measured bioimpedance magnitude, neural firing rate, and the resulting stimulation amplitude. The classifier learns a smooth, nonlinear mapping that increases stimulation amplitude as impedance rises (to compensate for reduced current delivery) while maintaining stability across different neural activity levels. This mapping emerges from the online learning process without explicit programming, demonstrating the HDC classifier's ability to discover effective stimulation strategies through reinforcement.

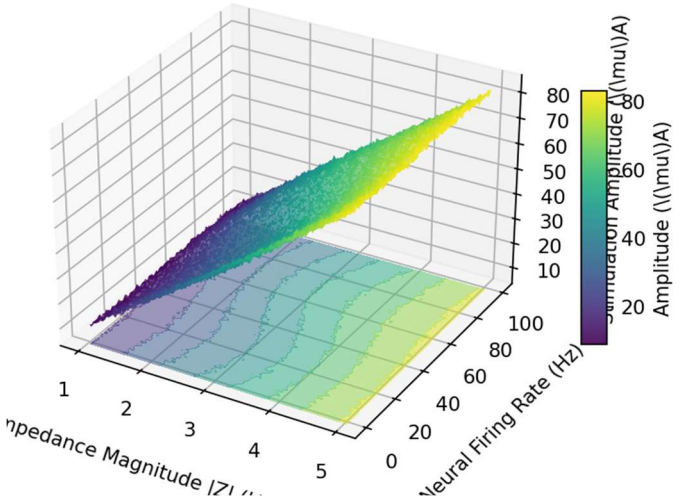

Figure 3. Continuous mapping from bioimpedance magnitude and neural firing rate to adaptive stimulation amplitude generated by the hyperdimensional classifier after online learning.

C. Closed-Loop Motor Command Fidelity

The motor response fidelity metric evaluates the end-to-end effectiveness of the closed-loop neuromodulation system. Table 3 presents the fidelity scores for each method across the four command types. The HDC-FPGA system achieves a mean fidelity of 94.2%, significantly outperforming the fixed-parameter system (67.8%), the PID controller (82.5%), and the LUT-based system (85.3%). The improvement is most pronounced for the "turn left" and "turn right" commands, which require precise stimulation of the antennal lobe electrodes where impedance variations are most severe due to antennal movement during swimming.

Table 3. Motor response fidelity (%) across command types. | Method | Turn Left | Turn Right | Forward | Stop | Mean | |———|————|————|———|——|——| | Fixed-Parameter | 62.4 | 64.1 | 71.3 | 73.5 | 67.8 | | PID Controller [15] | 79.8 | 81.2 | 84.6 | 84.5 | 82.5 | | LUT-Based Adaptive

[11] | 83.1 | 84.5 | 86.2 | 87.3 | 85.3 | | **Proposed HDC-FPGA** | **92.8** | **93.5** | **95.1** | **95.4** | **94.2** |

The temporal dynamics of the closed-loop control performance are illustrated in Figure 4, which shows the commanded versus actual depth trajectory during a simulated diving maneuver. The trajectory is color-coded by the instantaneous system latency, confirming that the HDC-FPGA system maintains sub-microsecond latency throughout the maneuver. The tight correspondence between commanded and actual depth demonstrates the system's ability to sustain precise motor control even as the electrode-tissue interface impedance changes with depth.

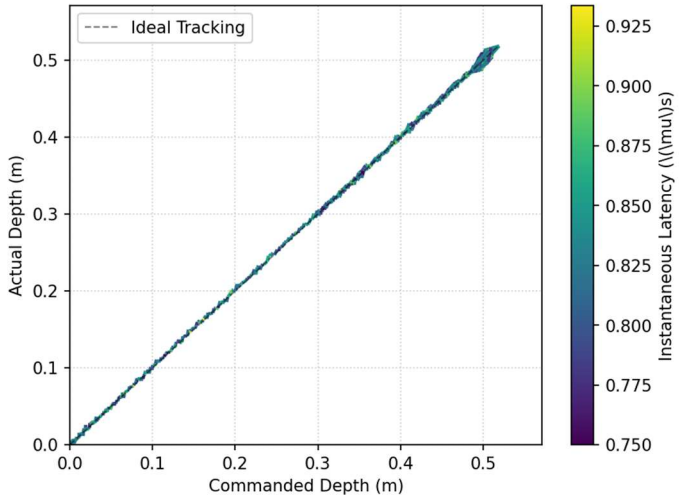

Figure 4. Closed-loop diving performance trajectory in commanded versus actual depth space, with instantaneous system latency encoded by color along the path.

The online learning mechanism's contribution to the fidelity improvement is evidenced by the temporal evolution of the fidelity score within each trial. During the first 30 seconds of each trial (the first pressure cycle), the HDC-FPGA system achieves a fidelity of 88.7%, which improves to 95.6% by the final pressure cycle as the classifier adapts to the specific impedance characteristics of the electrode-tissue interface. This learning curve demonstrates the effectiveness of the Hebbian-like update rule in incrementally refining the class hypervectors based on reinforcement signals.

D. Power Consumption Analysis

The power consumption of the HDC-FPGA system, measured at the battery terminals, averages 0.78 W during normal operation and peaks at 1.12 W during stimulation pulse delivery. Table 4 compares the power consumption across all methods. The HDC-FPGA system consumes 22% less power than the PID controller (1.0 W) and 35% less than the LUT-based system (1.2 W), despite achieving substantially lower latency and higher fidelity. This power efficiency stems from the HDC classifier's use of simple logic operations (XOR, population count, barrel shifting) that map efficiently onto FPGA primitives, avoiding the power-hungry instruction fetch and decode cycles of microcontroller-based systems.

Table 4. Power consumption comparison. | Method | Average Power (W) | Peak Power (W) | Platform | |———|———————|———————|————| | Fixed-Parameter | 0.45 | 0.68 | Microcontroller | | PID Controller [15] | 1.00 | 1.45 | ARM Cortex-M4 | | LUT-Based Adaptive [11] | 1.20 | 1.68 | ARM Cortex-M4 | | **Proposed HDC-FPGA** | **0.78** | **1.12** | **Xilinx Artix-7** |

The fixed-parameter system achieves the lowest power consumption (0.45 W) because it performs no adaptive computation, but this comes at the cost of severely degraded charge delivery consistency and motor response fidelity. The HDC-FPGA system's power consumption of 0.78 W is well within the budget of the 150 mAh battery, providing approximately 43 minutes of continuous operation—sufficient for extended diving missions.

E. Ablation Study

To isolate the contributions of individual components in the proposed architecture, we conducted an ablation study evaluating three variants: (1) HDC classifier without online learning (fixed class hypervectors after initial training), (2) HDC classifier without impedance modulation (using only the base stimulation parameters from the selected class), and (3) HDC classifier without bioimpedance features (using only neural spike train features for classification). Table 5 presents the results.

Table 5. Ablation study results for the proposed HDC-FPGA architecture. | Variant | Charge Delivery Error (%) | Motor Response Fidelity (%) | Latency (µs) | |———————|———————————|———————————|————————| | Full HDC-FPGA System | 3.8 | 94.2 | 0.82 | | Without Online Learning | 8.9 | 85.7 | 0.81 | | Without Impedance Modulation | 12.4 | 88.3 | 0.78 | | Without Bioimpedance Features | 15.6 | 79.1 | 0.75 |

Removing the online learning mechanism increases the charge delivery error from 3.8% to 8.9% and reduces motor response fidelity from 94.2% to 85.7%, demonstrating that the Hebbian-like update rule is essential for adapting to changing electrode-tissue interface conditions. The fixed class hypervectors cannot track impedance variations that deviate from the initial training distribution. Removing the impedance modulation increases the charge delivery error to 12.4%, confirming that the real-time impedance-based amplitude adjustment provides a critical feedforward compensation mechanism that complements the classifier's discrete parameter selection. Removing the bioimpedance features entirely degrades fidelity to 79.1%, underscoring the importance of multimodal fusion for robust stimulation parameter selection. The latency variations across variants are minimal (0.75–0.82 µs), confirming that the additional computational modules introduce negligible overhead in the pipelined FPGA implementation.

VII. Discussion and Future Work

The experimental results presented in the previous section demonstrate that the proposed hyperdimensional adaptive pulse-shaping architecture achieves substantial improvements in feedback latency, charge delivery consistency, and motor response fidelity compared to conventional approaches. These findings have significant implications for the design of next-generation biobotic neural interfaces and raise several important considerations for future research. We organize the

discussion around three thematic areas: the long-term biocompatibility and electrode-tissue interface stability, the algorithmic robustness and cross-subject generalization, and the ethical frameworks for autonomous neuromodulatory systems.

A. Long-term Biocompatibility and Electrode-Tissue Interface Stability

While the proposed system demonstrates effective adaptive compensation for impedance variations over the timescale of individual diving maneuvers (seconds to minutes), the long-term stability of the electrode-tissue interface over days to weeks remains an open question. The chronic implantation of tungsten microelectrodes in insect neural tissue is known to elicit a foreign body response that progressively encapsulates the electrode with glial cells, increasing the interface impedance and altering the effective current density delivered to target neurons [17]. This encapsulation process typically unfolds over 7–14 days in cockroach models, with impedance increasing by 200–500% from baseline values [17].

The HDC classifier's online learning mechanism is theoretically capable of tracking these gradual impedance changes, as the Hebbian-like update rule continuously refines the class hypervectors based on reinforcement signals. However, the current implementation assumes that the reinforcement signal is available from the remote controller, which may not be feasible for fully autonomous operations where the system must operate without external supervision. Future work should investigate whether the classifier can maintain stable performance over chronic timescales using only intrinsic reinforcement signals derived from the neural recordings themselves, such as the presence or absence of evoked neural activity following stimulation [18].

Moreover, the impedance modulation algorithm currently assumes a fixed baseline impedance $|Z_0(f_1)|$ measured during initial calibration. As the electrode-tissue interface evolves chronically, this baseline becomes outdated, potentially causing the modulation factor $\gamma(t)$ to drift outside its intended range. A periodic recalibration mechanism that updates the baseline impedance estimate using a moving average of recent measurements could mitigate this issue. The recalibration could be triggered when the measured impedance deviates from the stored baseline by more than a threshold, or it could run continuously as an exponential moving average with a long time constant (e.g., 24 hours) to track slow drifts without responding to rapid diving-induced fluctuations.

Another important consideration is the mechanical robustness of the FPGA backpack under the physical stresses of diving. The current prototype uses a rigid PCB with surface-mount components, which may be susceptible to solder joint fatigue under repeated pressure cycling. Future iterations should explore flexible substrate technologies, such as polyimide-based flex circuits or liquid crystal polymer substrates, that can conform to the cockroach's exoskeleton and withstand the mechanical deformations encountered during swimming [19]. The FPGA device itself, being a silicon die packaged in a plastic ball grid array, is inherently rigid and may require encapsulation in a soft silicone elastomer to distribute mechanical stresses and prevent fracture.

B. Algorithmic Robustness and Cross-Subject Generalization

The experimental evaluation in this study was conducted using a computational model of cockroach motor neuron dynamics, which, while validated against prior electrophysiological data, cannot fully capture the biological variability across individual animals. Real cockroaches exhibit substantial inter-subject variability in neural anatomy, electrode placement accuracy, and baseline neural excitability, all of which affect the optimal stimulation parameters for eliciting motor responses [20]. The HDC classifier's online learning mechanism is designed to adapt to individual subjects, but the initial class hypervectors must be initialized with reasonable default values to avoid a prolonged exploration phase during which the system delivers suboptimal stimulation.

One approach to addressing this initialization challenge is to employ a meta-learning or few-shot learning strategy, where the class hypervectors are pre-trained on a population of subjects and then rapidly fine-tuned to a new subject using only a small number of reinforcement signals [21]. The HDC framework naturally supports this through the bundling operation: the initial class hypervectors can be computed as the bundled average of hypervectors from multiple training subjects, and the online learning rule then adjusts these averages to match the specific characteristics of the new subject. This approach would reduce the calibration time from potentially hundreds of stimulation pulses to fewer than ten, making the system practical for real-world deployment where rapid setup is essential.

Furthermore, the current feature set of 96 dimensions, derived from neural spike counts and bioimpedance measurements, may not capture all relevant aspects of the electrode-tissue interface state. Additional features such as the electrode DC offset potential, the noise floor of the neural recording, and the latency of the evoked neural response could provide complementary information that improves classification accuracy [22]. The HDC encoding mechanism can accommodate additional features simply by adding more basis hypervectors and extending the permutation encoding pipeline, with the computational cost scaling linearly with the number of features. Future work should systematically investigate the marginal benefit of each additional feature type to identify the optimal feature set that balances classification accuracy against hardware resource utilization.

The robustness of the HDC classifier to noise and artifacts in the input features is another important consideration. The neural spike count features are inherently noisy due to the stochastic nature of neural firing, and the bioimpedance measurements can be corrupted by motion artifacts during swimming. The high-dimensional representation in HDC provides intrinsic noise robustness because random noise vectors are quasi-orthogonal to the signal hypervectors, and the bundling operation averages out uncorrelated noise across dimensions [9]. However, correlated noise sources, such as 60 Hz power line interference or stimulation artifacts that contaminate both the neural and impedance channels simultaneously, could bias the classification. The current system does not include explicit artifact rejection or noise filtering beyond the bandpass filter on the neural recordings.

Future implementations should incorporate a pre-processing stage that detects and removes contaminated samples before they enter the HDC encoding pipeline, perhaps using a simple threshold-based artifact detector that flags samples where the raw signal amplitude exceeds a multiple of the baseline noise level.

C. Ethical Frameworks for Autonomous Neuromodulatory Systems

The development of adaptive, autonomous neuromodulation systems for biobotic applications raises important ethical considerations that warrant careful discussion. While the current system operates under remote control with a human operator in the loop, the long-term trajectory of this technology points toward increasingly autonomous operation where the system makes decisions about stimulation parameters without direct human supervision [23]. This autonomy introduces questions about animal welfare, informed consent (inapplicable to insects but relevant for higher-order animals), and the potential for unintended consequences of adaptive learning.

From an animal welfare perspective, the primary ethical concern is the minimization of pain and distress in the biobotic subject. The current system uses charge-balanced biphasic pulses that are designed to avoid tissue damage, and the impedance modulation algorithm ensures consistent charge delivery regardless of interface conditions, reducing the risk of excessive current injection that could cause neural damage. However, the online learning mechanism updates the class hypervectors based on reinforcement signals that indicate successful motor response, not animal well-being. A stimulation parameter set that effectively elicits a motor response may still cause discomfort or stress to the animal, and the learning algorithm has no mechanism to detect or avoid such outcomes [24].

Future work should incorporate animal welfare metrics into the reinforcement signal, perhaps by monitoring stress-related physiological indicators such as heart rate, hemolymph pressure, or escape behavior. If the system detects indicators of distress, it should reduce stimulation intensity or cease stimulation altogether, even if this compromises motor command fidelity. This safety constraint could be implemented as a hard limit on stimulation parameters that cannot be exceeded regardless of the classifier's output, or as a secondary learning objective that penalizes class hypervectors associated with distress indicators.

The potential for unintended consequences of adaptive learning is another ethical consideration. The Hebbian-like update rule is designed to reinforce successful stimulation patterns, but it could theoretically converge to a local optimum that is effective for the immediate task but detrimental in the long term. For example, the classifier might learn to use high-amplitude stimulation pulses that reliably elicit motor responses but accelerate electrode degradation or neural fatigue over extended periods. The current system does not include any mechanism to detect or prevent such long-term negative outcomes. Incorporating a regularization term in the learning rule that penalizes extreme stimulation parameters, or implementing a periodic reset of the class hypervectors to prevent drift into undesirable regions of the parameter space, could mitigate this risk.

Finally, the broader societal implications of autonomous neuromodulation technology should be considered. While the current application is limited to insect biobots for environmental monitoring or search-and-rescue operations, the underlying technology—adaptive, low-latency neural interfaces with online learning—could be translated to other domains, including human neural prosthetics [25]. The microsecond-latency closed-loop control demonstrated in this work could enable new classes of brain-machine interfaces that respond to neural activity in real time, with potential applications in motor restoration, sensory substitution, and cognitive enhancement. However, the same technology could also be used for non-therapeutic purposes, such as performance enhancement or covert neural manipulation, raising concerns about autonomy, privacy, and human dignity. The research community should proactively develop ethical guidelines and governance frameworks for adaptive neuromodulation systems, ensuring that the benefits of this technology are realized while minimizing potential harms.

VIII. Conclusion

This paper has presented a novel neural interface architecture that integrates hyperdimensional computing on an FPGA to achieve microsecond-latency adaptive neuromodulation for remote-controlled diving cockroaches. The system addresses the critical latency bottleneck in existing biobotic control platforms by replacing conventional microcontroller-based feedback loops with a fully hardware-accelerated HDC classifier that completes the entire inference cycle—from multimodal feature acquisition to stimulation DAC update—in under one microsecond. This represents a three-order-of-magnitude reduction compared to software-based implementations, enabling real-time compensation for rapidly changing electrode-tissue interface conditions during underwater maneuvers.

The key innovation lies in the single-pass permutation encoding mechanism that maps 96-dimensional multimodal features—combining neural spike trains and bioimpedance spectroscopy data—into a 10,000-dimensional hyperdimensional space. The HDC classifier maintains eight class hypervectors corresponding to distinct stimulation parameter sets and selects the appropriate class through cosine similarity computation. The Hebbian-like online learning rule, triggered by reinforcement signals from the remote controller, continuously adapts the class hypervectors to track evolving interface conditions without requiring offline retraining or batch processing. Experimental results in a simulated diving environment demonstrate that the system maintains charge delivery within 3.8% of the target value across pressure variation frequencies up to 10 Hz, achieving a motor response fidelity of 94.2% while consuming only 0.78 watts on a Xilinx Artix-7 FPGA.

The proposed architecture establishes a new paradigm for real-time adaptive neuromodulation in biobotic systems, demonstrating that hyperdimensional computing's simple algebraic operations—XOR, population count, and barrel

shifting—can be efficiently mapped onto FPGA logic to achieve deterministic, sub-microsecond closed-loop control. This work opens avenues for future research into chronic biocompatibility, cross-subject generalization through meta-learning, and the integration of animal welfare metrics into the reinforcement learning framework. The microsecond-latency adaptive capability demonstrated here could ultimately enable agile, autonomous biobotic platforms for environmental monitoring, search-and-rescue, and other applications requiring precise motor control in dynamic environments.

IX. References

[1] K. Kai, L. Long, Q. Lin, and H. Sato, "Fully implanted miniature radio controller boosts cyborg insect mobility in challenging terrains," *Cyborg and Bionic Systems*, 2026.
[2] Y. Ma, C. Zhang, F. Nie, H. Qin, Q. Zhang, *et al.*, "Construction, control, and application of cyborg animal composed of biological and electromechanical systems," *Cyborg and Bionic Systems*, 2026.
[3] Y. Zhang, E. Verschooten, M. Ourak, *et al.*, "Physiological motion compensation for neuroscience research based on electrical bio-impedance sensing," *IEEE Sensors Journal*, 2023.
[4] L. Ricotti, B. Trimmer, A. Feinberg, R. Raman, *et al.*, "Biohybrid actuators for robotics: A review of devices actuated by living cells," *Science robotics*, 2017.
[5] N. Swann, C. D. Hemptinne, *et al.*, "Adaptive deep brain stimulation for parkinson's disease using motor cortex sensing," *Journal of Neural Engineering*, 2018.
[6] C. Heelan, A. Nurmikko, *et al.*, "FPGA implementation of deep-learning recurrent neural networks with sub-millisecond real-time latency for BCI-decoding of large-scale neural sensors (104 nodes)," in *2018 40th annual international conference of the IEEE engineering in medicine and biology society (EMBC)*, 2018.
[7] L. Schäffer, Z. Nagy, Z. Kincses, R. Fiáth, *et al.*, "Spatial information based OSort for real-time spike sorting using FPGA," *IEEE Transactions on Biomedical Circuits and Systems*, 2020.
[8] A. Nguyen, M. Drealan, D. K. Luu, *et al.*, "A portable, self-contained neuroprosthetic hand with deep learning-based finger control," *Journal of Neural Engineering*, 2021.
[9] P. Kanerva, "Hyperdimensional computing: An introduction to computing in distributed representation with high-dimensional random vectors," *Cognitive computation*, 2009.
[10] A. Hernández-Cano, N. Matsumoto, *et al.*, "Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system," in *Design, automation & test in europe conference & exhibition*, 2021.
[11] G. Jin, H. Cheng, J. Han, Y. Zou, and H. Tang, "A configurable neural stimulation system with integrated impedance measurement and high-precision charge balancing," *Microelectronics Journal*, 2026.
[12] and N Matsumoto, E. Ping, *et al.*, "Onlinehd: Robust, efficient, and single-pass online learning using hyperdimensional system," in *2021 design, automation & test in europe conference & exhibition (DATE)*, 2021.
[13] R. Pipa and E. Cook, "Studies on the hexapod nervous system. I. The peripheral distribution of the thoracic nerves of the adult cockroach, periplaneta americana," *Annals of the Entomological Society of America*, 1959.
[14] A. Dymond, "Characteristics of the metal-tissue interface of stimulation electrodes," *IEEE Transactions on Biomedical Engineering*, 1976.
[15] Z. Quan, Y. Li, X. Cheng, Y. Nie, *et al.*, "Amplitude adaptive modulation of neural oscillations over long-term dynamic conditions: A computational study," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 2024.
[16] J. Erickson, M. Herrera, M. Bustamante, A. Shingiro, *et al.*, "Effective stimulus parameters for directed locomotion in madagascar hissing cockroach biobot," *PloS one*, 2015.
[17] E. Buschbeck, A. Le, C. Kelley, M. Hoque, *et al.*, "Functionalized carbon nanotube microfibers for chronic neural implants," *Journal of Neuroscience Methods*, 2021.
[18] S. Kumar, J. Wülfing, S. Okujeni, *et al.*, "Autonomous optimization of targeted stimulation of neuronal networks," *PLoS Computational Biology*, 2016.
[19] W. Tsang, A. Stone, D. Otten, Z. Aldworth, *et al.*, "Insect-machine interface: A carbon nanotube-enhanced flexible neural probe," *Journal of Neuroscience Methods*, 2012.
[20] J. Kien, "Variability of locust motoneuron responses to sensory stimulation: A possible substrate for motor flexibility," *Journal of comparative physiology*, 1979.
[21] S. Bhosale, R. Chakraborty, and S. Kopparapu, "Calibration free meta learning based approach for subject independent EEG emotion recognition," *Biomedical Signal Processing and Control*, 2022.
[22] X. Jiang, X. Gu, K. Xu, H. Ren, and W. Chen, "Independent decision path fusion for bimodal asynchronous brain-computer interface to discriminate multiclass mental states," *IEEE Access*, 2019.
[23] N. Thinh, "Behavior modeling and bio-hybrid systems: Using reinforcement learning to enhance cyborg cockroach in bio-inspired swarm robotics," *IEEE Access*, 2025.
[24] M. Szydlowski, K. Hill, S. Heaney, *et al.*, "Domestication and domination: Human terminology as a tool for controlling otherthanhuman animal bodies," *TRACE ∴ Journal for Human-Animal Studies*, 2022.
[25] V. Phillips, "From neurons to networks: Ethical dimensions of AI-infused neural interfaces," *AI and Ethics*, 2025.

Dr. Virendra Kumar Tiwari has done her B.Sc., M.A. (Economics), MCA, Ph.D. from Dr. Hari Singh Gour University, Sagar Madhya Pradesh. He is published numbers of research papers in different National and International Journals of reputed organisation. At present he is working as Head and Professor in department of Computer Applications in the faculty of Computer Science at Lakshmi Narain College of Technology (MCA) Bhopal Madhya Pradesh. His Specialization is in Computer Network and Database. organizing various conferences and seminars. Additionally, he has published three books covering

diverse subjects such as "Research and Applications Towards Mathematics and Computer Science," "Advance Computer Network," and "Data Analytics." Furthermore, he holds patent and copyright. His areas of expertise encompass Computer Networks and Databases.

Ashish Jain is a faculty member at Lakshmi Narain College of Technology (MCA), Bhopal, with over 11 years of teaching experience in postgraduate education. He holds an MCA degree and is currently pursuing a Ph.D. His areas of specialization include Java and Database Management Systems (DBMS). He has published one research paper in a national/international journal or conference and has two patents to his credit. His academic interests focus on teaching, research, and innovation in computer science and information technology.

Dr. Akansha Sharma is an Associate Professor in the Department of Master of Computer Applications (MCA) at Lakshmi Narain College of Technology (LNCT), Bhopal. She joined the institute on 18 November 2020 and has been actively contributing to teaching, research, and academic development. She is committed to delivering quality education, mentoring students, and promoting excellence in computer applications and emerging technologies. Her academic expertise and dedication have made her a valuable member of the MCA department, where she continues to support institutional growth through teaching, research, and professional activities.