# Hyperdimensional Computing for Acoustic and Proprioceptive Terrain Classification

Vishruti Ranjan Kandala Savitha Viswanadh Kanewala Udaree Chathurangee Hiranthika

## Abstract

**Accurate terrain classification is critical for autonomous robots operating in dynamic environments, as safe navigation, adaptive locomotion and stability are essential. Existing vision-based approaches often struggle under visually degraded conditions and high computational and power demands. To address these limitations, we present a low-power terrain classification framework based on hyperdimensional computing (HDC), along with the fusion of acoustic and proprioceptive sensing. In contrast to conventional machine learning methods that rely on high computational resources, HDC provides a brain-inspired, energy-efficient alternative well-suited for edge deployment. Our results demonstrate promising classification accuracy (of up to 100%) with a low memory footprint (34.69 KB) and low power profile (4.9 mW), paving the way for more adaptable and efficient robotic systems. Our code repo is open-sourced online[1].**

## 1 Introduction

Accurate terrain classification is essential for autonomous robots operating in complex environments as they need to navigate various kinds of surfaces while ensuring safe navigation, stability, and energy-efficient locomotion [15]. For example, autonomous home service robots need to navigate on diverse surfaces such as wood, tile, carpet, metal etc., to perform their operations, and they need to be able to adjust their speeds based on the contact surfaces in order to avoid slippage and accidents. Current solutions for terrain classification involve cameras and vision-based models deployed on embedded systems with high power demands.

Traditional vision-based terrain classification methods suffer from several limitations: they are computationally expensive, power-intensive, and unreliable in visually degraded conditions such as low light, fog, or occlusions [14]. For example, reflective floors can mislead vision-based sensors, causing the robot to misinterpret reflections as actual obstacles or even fail to recognize real obstacles [22]. Such misinterpretations can result in navigation errors or collisions due to misclassification of terrain. Further, robots operating on slippery indoor surfaces may fail to detect the lack of traction, leading to slips and potential damage [16]. This raises a key concern about the reliability of such methods in real-world applications, particularly for robots operating in domestic environments, medical facilities, industrial and commercial spaces. These limitations pose significant challenges for resource-constrained embedded systems, often making them unsuitable for low-power deployments [5].

To overcome these challenges, multimodal sensing, specifically acoustic [23], [14] and proprioceptive sensing [21], [1], offer a promising alternative as they do not rely on external lighting conditions, making them more resilient to occlusions and variable environments. Acoustic sensors capture sounds of terrain interaction, while proprioceptive sensors measure forces and vibrations experienced by robot actuators. Unlike vision-based methods, these modalities provide continuous, environment-independent feedback, making them highly suitable for real-time terrain classification.

Energy efficiency is another critical factor in deploying machine learning-based terrain classification models on embedded robotic platforms. Previous studies have explored audio classification using traditional machine learning models such as support vector machines (SVMs) and deep neural networks (DNNs). While these traditional deep learning models offer reasonable accuracy, they require significant computational resources and energy, often necessitating GPU acceleration, which is impractical for battery-powered robots. In this work, we explore Hyperdimensional Computing (HDC), a brain-inspired computational paradigm that offers simplicity, robustness, and energy efficiency, making it a promising alternative for embedded classification tasks [7].

Our hypothesis is that combining acoustic and proprioceptive sensing with Hyperdimensional Computing (HDC) enables efficient and accurate terrain classification for autonomous robots. Unlike conventional deep learning approaches, HDC leverages binary (or balanced ternary) operations and memory-centric computing, making it inherently suitable for low-power embedded systems. Its lightweight, associative learning model supports real-time decision-making with minimal energy consumption while maintaining competitive accuracy [8], [13].

We propose an HDC-based multimodal terrain classification framework that fuses acoustic and proprioceptive data to enhance robotic autonomy in energy-constrained environments. This approach advances terrain perception by enabling adaptive, reliable classification and contributes to safer and more intelligent navigation in real-world scenarios.

## 2 Related Work & Background

This section reviews key concepts and prior work in terrain classification for robots, the use of acoustic and proprioceptive sensing, and the fundamentals of HDC.

### 2.1 Terrain Classification for Robots

Terrain classification plays an important role in robotic systems, especially in applications involving navigation, mobility, and adaptive control in unstructured environments. In planetary exploration, search-and-rescue, and agricultural automation, the ability to reliably classify ground materials enables robots to adjust their locomotion strategies, plan safer paths, and optimize energy consumption. Recent surveys [15] highlight that visual sensing remains the most common approach to terrain classification. However, its limitations, such as sensitivity to lighting conditions and occlusions, have driven interest in complementary sensing modalities, including vibration

[1] github.com/sevenseasofbri/terrain-classifier/

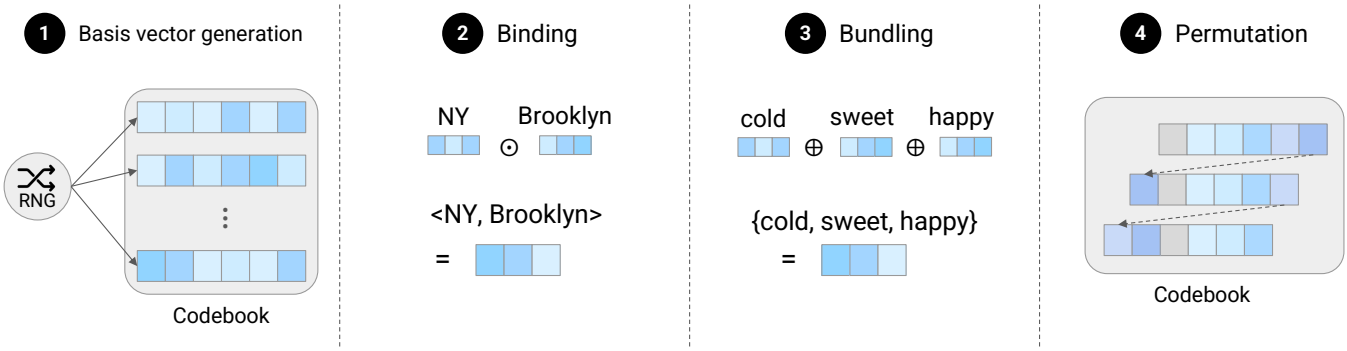



**Figure 1: Key operations of Binary Spatter Code. (1) Basis vectors are generated for every letter. (2) Binding of data creates a record. (3) Bundling of words creates a set. (4) Permutation is applied to create hypervectors on-the-fly. This figure is adapted from HyperCam [13]**

and force feedback. In parallel, there has been a notable shift toward data-driven methods, with machine learning models such as Support Vector Machines (SVMs), Artificial Neural Networks (ANNs), and ensemble techniques such as Probabilistic Neural Networks (PNNs) gaining popularity. This growing body of work supports a trend toward multimodal sensing strategies that integrate diverse data sources to improve robustness and classification accuracy.

## 2.2 Multimodal Perception: Audio and Proprioceptive Sensing

*2.2.1 Audio Sensing.* Audio signals provide an information-rich modality for recognizing a material's properties. When a robot or vehicle interacts with terrain (walking, rolling, tapping, etc.), acoustic signatures can be captured and analysed to differentiate materials such as grass, sand, or even metal. Early works such as Libby and Stentz (2012) [14] and Tesfay et al. (2015) [19] demonstrated that sound alone can distinguish between terrain types under varying environmental conditions. These systems typically transform raw audio into spectral features using Fast Fourier Transform (FFT) or Mel-frequency cepstral coefficients (MFCCs), and apply classifiers such as Support Vector Machines (SVMs) or feed-forward neural networks, achieving classification accuracies of up to 92%. In terms of sensing hardware, a practical distinction arises between high-fidelity analog microphones and digital sensors such as MEMS microphones. Analog microphones can capture a wider dynamic range and finer acoustic details, but they often require dedicated analog-to-digital converters and are more sensitive to environmental noise. Digital MEMS microphones, by contrast, integrate signal conversion and are well suited for embedded systems due to their compact size, low power consumption, and simplified data acquisition. Their ease of integration makes them ideal for mobile robotic applications, such as in our work.

*2.2.2 Proprioceptive Sensing.* Proprioceptive sensors, including accelerometers, force sensors, and tactile arrays, capture the physical interaction forces between the robot and its environment. In terrain classification, these sensors offer a complementary modality to exteroceptive sensing like vision. Among these, vibration signals are particularly good at revealing of surface texture and stiffness. For instance, Bai et al. (2019) [2] demonstrated the use of vibration data to classify terrain types, including cement, brick, and soil, employing a deep multilayer perceptron for high-accuracy classification. Proprioceptive sensing has also proven effective in challenging domains such as planetary exploration. Brooks and Iagnemma (2012) [4] applied self-supervised learning to integrate vibration and visual data on a planetary rover, allowing the system to adapt terrain recognition models based on its own experience. These studies highlight the utility of proprioceptive cues in environments where vision alone may be insufficient or unreliable.

## 2.3 Hyperdimensional Computing (HDC)

To perform on-board inference with multimodal data, we use HDC. It is a framework that is lightweight, noise-tolerant and suited for resource-constrained robotic platforms. Inspired by the properties of high-dimensional spaces, HDC represents data as binary or bipolar hypervectors and performs operations like binding, bundling, and permutation using simple arithmetic or bitwise logic [12]. From its inception, HDC has been studied for its theoretical capacity [20], incremental learning potential [7], and hardware efficiency [9]. These properties make it ideal for real-time embedded classification tasks. In audio and tactile processing, HDC has demonstrated competitive performance to deep learning models with significantly lower memory and energy usage. For example, Yun et al. [24] showed that HDC can classify acoustic events with more than 80% energy savings and minimal accuracy loss. Key to this efficiency is HDC's ability to encode temporal and spectral features into compact representations that preserve signal characteristics despite noise or variation.

This work adopts the Binary Spatter Code (BSC) approach [11], wherein each component of a hyper-vector is binary. Operations on such binary hyper-vectors are computationally simple and energy-efficient, making them particularly well-suited for deployment on resource-constrained hardware platforms. The subsequent subsection, along with Figure 1, provides a detailed explanation of the BSC-based hyper-dimensional computing (BSC-HDC) operations.

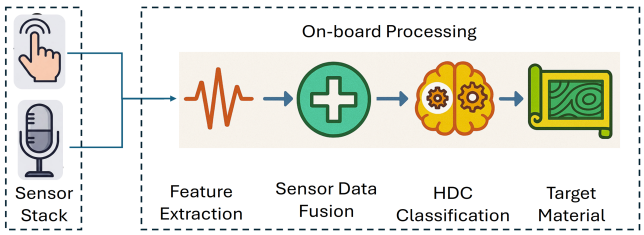



**Figure 2: System workflow of the proposed embedded system for terrain classification.**

## 2.4 Binary Spatter Code

Binary Spatter Code (BSC) uses high-dimensional binary vectors (hyper-vectors (HV)) and simple bitwise operations to encode and manipulate data efficiently.

*2.4.1 Basis Vector Generation.* Each symbol (e.g., feature name or value level) is assigned a random binary HV of dimension $D$ with bits drawn i.i.d. Bernoulli(0.5). These HVs are nearly orthogonal, ensuring distinct codes.

*2.4.2 Binding.* Binding ($\odot$) combines two HVs into a new one that is dissimilar to both inputs. In BSC, binding is implemented as bitwise XOR:

$$H_{\text{bind}} = H_a \odot H_b$$

This operation encodes relationships (e.g., feature name bound with its value).

*2.4.3 Bundling.* Bundling ($\oplus$) aggregates multiple HVs into a single representative HV by bit-wise majority vote. Given multiple $\{H_i\}_{i=1}^{n}$, the bundled HV $H_{\text{bundle}}$ is

$$H_{\text{bundle}}[j] = \begin{cases} 1 & \sum_i H_i[j] \geq \lceil n/2 \rceil, \\ 0 & \text{otherwise.} \end{cases}$$

This preserves common patterns across inputs.

*2.4.4 Permutation.* Permutation reorders bits in a HV (e.g., circular shift) to produce a new, dissimilar code. It is used to encode positional or order information without additional storage:

$$H_{\text{perm}} = \text{permute}(H, k) \quad \text{(shift by } k\text{)}.$$

*2.4.5 Distance Metric.* Similarity between HVs is measured by normalized Hamming distance:

$$\text{dist}(H_a, H_b) = \frac{1}{D} \sum_{j=1}^{D} \left| H_a[j] - H_b[j] \right|.$$

Lower distance implies higher similarity, which is used for classification by nearest-class HV lookup.

# 3 System Design

The goal of our system is to enable real-time on-board terrain classification that is low-power and not easily hindered by varying environmental conditions. To achieve this, we design and explore a lightweight sensing and inference pipeline based on complementary modalities - acoustic and proprioceptive sensing. A high-level overview of the system is shown in Figure 2, with the pipeline organized into five stages: Sensing, Feature Extraction, Sensor Fusion, HDC Classification, and Material Identification.

Our sensor stack uses audio and vibration to capture physical interactions between the robot and the terrain. Acoustic sensing captures the sound produced by the robot wheel interacting with different surfaces. Proprioceptive sensing detects surface-induced vibrations during locomotion.

These signals are recorded close to a robot wheel–terrain contact point to maximize signal precision. The microphone is mounted nearly 5–10 cm from the wheel to capture direct wheel interaction sounds, while the vibration sensor is mounted in a way that it contacts the surface and is perpendicular to it and aligns with the direction of motion. This orientation ensures that terrain-induced mechanical energy is accurately transmitted to the sensor.

The collected sensor data undergoes feature extraction, which is then fused and encoded into hyper-dimensional (HD) vectors to generate compact representations. Once the HD Classifier is trained, the full pipeline is deployed on the target microcontroller to perform on-board inference to identify terrain in real time based on sound and vibration data.

Details of the learning algorithm and hardware setup are presented in Sections § 4 and §5.

## 3.1 Sensor Selection

*3.1.1 Acoustic Sensing.* We use a digital MEMS microphone to capture audio of terrain interaction. Compared to analog mics, MEMS mics offer lower power consumption, integrated signal digitalization, compact size, and most importantly are easier to integrate into embedded platforms.

*3.1.2 Proprioceptive Sensing.* For proprioceptive sensing, we use a piezoelectric vibration sensor due to its passive sensing capabilities which provides a low-power solution compared to other active vibration sensors. It has a compact form factor and good sensitivity to surface texture variations. This sensor outputs a voltage proportional to mechanical strain caused by terrain-induced vibrations.

Both sensors are lightweight, energy-efficient, and support real-time signal acquisition, making them well-suited for edge deployment on mobile robotic platforms.

## 3.2 Wheel Design

For the robot wheel, we 3D-printed a specific design using Polylactic Acid (PLA) that amplifies the surface interactions over a standard rubber wheel. The wheel contains deep groves which can interact with the surfaces better in order to provide good sound quality for the data collection and live processing. Though the choice of PLA as the wheel material was based on availability, the rigid structure produces clear acoustic feedback. The design impacts both sensing and locomotion, meaning any classification pipeline must be co-tuned with the wheel properties.

# 4 Learning Algorithm

This section details the complete pipeline that maps raw proprioceptive audio & vibration signals to a predicted **material class** by means of the HDC paradigm. Figure 3 shows an overview of the

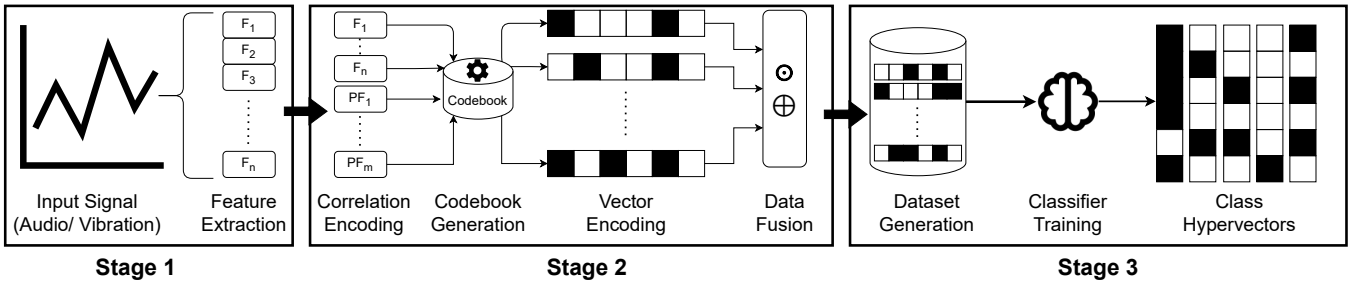



**Figure 3: Overview of the learning algorithm**

various stages that are part of the pipeline. The pipeline comprises three stages:

1. **Feature Extraction** – frame-level signal processing that yields a compact, discriminative descriptor for each modality (§ 4.1).
2. **Data Fusion and HD Encoding** – conversion of every scalar feature into a binary hypervector (HV), followed by feature–feature and cross-modality fusion (§ 4.2).
3. **HD Classifier Training and Inference** – online prototype learning with an OnlineHD [10] update rule and nearest-prototype decoding (§ 4.3).

Throughout, binding ($\odot$) is realised by bit-wise XOR, while bundling ($\oplus$) denotes majority-vote superposition of HVs.

## 4.1 Feature Extraction

Let $x[n]$ be a discrete-time audio or vibration signal of length $N_{\text{sig}}$. The signal is divided into $T$ overlapping frames of $N = 2048$ samples (15.625 kHz) with a hop length of $H = 512$. For each frame $t \in \{1, \ldots, T\}$, seven scalar features are computed:

**F1**: **Zero-Crossing Rate (ZCR)** – counts sign changes in the waveform, capturing rapid polarity flips characteristic of rough or granular surfaces.

$$\text{ZCR}_t = \frac{1}{2N} \sum_{n=1}^{N} |\text{sgn}(x_t[n]) - \text{sgn}(x_t[n-1])| \,. \tag{1}$$

**F2**: **Root-Mean-Square Energy (RMS)** – measures average signal power; higher values indicate harder, more energetic contacts.

$$\text{RMS}_t = \sqrt{\frac{1}{N} \sum_{n=1}^{N} x_t[n]^2} \,. \tag{2}$$

**F3**: **Temporal Entropy (TE)** – quantifies amplitude unpredictability; low entropy denotes periodic signals, high entropy reflects irregular interactions. Let $p_i$ be the histogram-bin probabilities, then:

$$\text{TE}_t = -\sum_{i=1}^{B} p_i \log_2 p_i. \tag{3}$$

**F4**: **Spectral Centroid (SC)** – the "centre of gravity" of the power spectrum; a higher centroid implies energy concentrated at higher frequencies.

$$\text{SC}_t = \frac{\sum_k f_k |X_t[k]|}{\sum_k |X_t[k]|} \,. \tag{4}$$

**F5**: **Spectral Roll-off (SR)** – frequency below which 85 % of spectral energy resides; separates damped from broadband signals.

$$\sum_{k=1}^{R} |X_t[k]| = 0.85 \sum_{k=1}^{N} |X_t[k]|. \tag{5}$$

**F6**: **Spectral Flatness (SF)** – ratio of geometric to arithmetic mean of the magnitude spectrum; high values denote noise-like, broadband textures.

$$\text{SF}_t = \frac{\exp\left(\frac{1}{N} \sum_k \ln |X_t[k]|\right)}{\frac{1}{N} \sum_k |X_t[k]|} \,. \tag{6}$$

**F7**: **Band-Energy Ratio (BR)** – compares mid-band (1–4 kHz) to low-band (100 Hz–1 kHz) energy, highlighting spectral emphasis linked to surface resonance.

$$\text{BR}_t = \frac{\sum_{k \in \text{mid}} |X_t[k]|}{\sum_{k \in \text{low}} |X_t[k]|} \,. \tag{7}$$

Frame values are averaged over $T$ frames to yield one scalar per feature and modality:

$$\bar{f} = \frac{1}{T} \sum_{t=1}^{T} f_t. \tag{8}$$

These features are essential in determining certain qualities from the sensor signals. These capabilities are denoted in the Table 1. Also, the time-domain signals are converted to frequency-domain using Cooley-Tukey's FFT algorithm [3].

**Table 1: Feature Discrimination Capabilities**

| Feature | Distinguishes |
|---|---|
| ZCR | Smooth vs. rough textures |
| RMS | Soft vs. hard contact vibrations |
| TE | Periodic vs. irregular signal patterns |
| SC | Low-frequency vs. high-frequency emphasis |
| SR | Concentrated vs. spread spectral energy |
| SF | Tonal clarity vs. noisy content |
| BR | Dominant mid-band vs. low-band energy |

## 4.2 Data Fusion and HD Encoding

This stage converts every feature vector $\bar{\boldsymbol{f}} = [\bar{f}_1, \ldots, \bar{f}_7]$ into a single $D$-bit HV $\boldsymbol{H} \in \{0,1\}^D$ and involves four major steps.

*4.2.1 Codebook Design.* For each modality we create two codebooks of random HVs:

- $C_{\text{feat}} : \{1, \ldots, 7\} \to \{0,1\}^D$ (feature identity HVs),
- $C_{\text{val}} : \{0, \ldots, 2^B - 1\} \to \{0,1\}^D$ (quantisation-level HVs).

The first entry in each codebook is sampled i.i.d. from Bernoulli(0.5); subsequent entries are generated by a random circular permutation $\rho^k$ to ensure near-orthogonality while controlling pairwise Hamming distances. The value of $B$ can be chosen according to the sensor's ADC precision (10-bit or 12-bit).

*4.2.2 Scalar Encoding.* Every normalised scalar $\bar{f}$ is linearly quantised to an integer $q \in \{0, \ldots, 2^B - 1\}$ and bound with its feature HV:

$$\boldsymbol{h}_f = C_{\text{feat}}(f) \;\odot\; C_{\text{val}}(q). \tag{9}$$

*4.2.3 Correlation Encoding.* To capture complementary information, four empirically selected feature pairs are created:

$$\begin{aligned} \boldsymbol{p}_1 &= \boldsymbol{h}_{\text{ZCR}} \odot \boldsymbol{h}_{\text{TE}}, & \boldsymbol{p}_2 &= \boldsymbol{h}_{\text{RMS}} \odot \boldsymbol{h}_{\text{SR}}, \\ \boldsymbol{p}_3 &= \boldsymbol{h}_{\text{SF}} \odot \boldsymbol{h}_{\text{SC}}, & \boldsymbol{p}_4 &= \boldsymbol{h}_{\text{RMS}} \odot \boldsymbol{h}_{\text{BR}}. \end{aligned} \tag{10}$$

*4.2.4 Sample-Level Bundling.* The modality-specific HV for sample $i$ is obtained by bundling the seven scalar HVs and four pair HVs:

$$\boldsymbol{H}_i^{(m)} = \bigoplus_f \boldsymbol{h}_f \;\oplus\; \bigoplus_{j=1}^{4} \boldsymbol{p}_j, \qquad m \in \{\text{audio}, \text{vibration}\}. \tag{11}$$

The two modality HVs are then bundled:

$$\boldsymbol{H}_i = \boldsymbol{H}_i^{(\text{audio})} \;\oplus\; \boldsymbol{H}_i^{(\text{vibration})}. \tag{12}$$

## 4.3 HD Classifier

An HD classifier of dimensionality $D = 10\,000$ is used. Let $\boldsymbol{P}_c \in \mathbb{Z}^D$ be the integer accumulator for class $c$.

*4.3.1 Online Training.* For each training epoch and sample $(\boldsymbol{H}_i, y_i)$:

(1) **Prediction**: compute normalised Hamming distance to every binarised prototype and predict:

$$\hat{y} = \arg\min_c \operatorname{dist}\big(\boldsymbol{H}_i, \operatorname{sgn}(\boldsymbol{P}_c)\big). \tag{13}$$

(2) **Prototype Update**: if $\hat{y} \neq y_i$, remove the current sample from the predicted class's accumulator and add it to the actual class's accumulator:

$$\boldsymbol{P}_{y_i} \leftarrow \boldsymbol{P}_{y_i} + \boldsymbol{H}_i, \qquad \boldsymbol{P}_{\hat{y}} \leftarrow \boldsymbol{P}_{\hat{y}} - \boldsymbol{H}_i. \tag{14}$$

(3) **Binarisation**: after each epoch, binarise the vector:

$$\boldsymbol{C}_c[j] = \begin{cases} 1, & \text{if } \boldsymbol{P}_c[j] \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

*4.3.2 Inference.* For an unseen sample, Eqs. (9)–(12) yield $\boldsymbol{H}_{\text{test}}$. The predicted class is

$$\widehat{y} = \arg\min_c \operatorname{dist}\big(\boldsymbol{H}_{\text{test}}, \boldsymbol{C}_c\big). \tag{16}$$

The binary OnlineHD [10] approach offers constant-time inference, single-pass incremental training, and strong robustness to sensor noise.

## 4.4 Memory Footprint

The HD framework is edge-deployment friendly because both codebooks and prototypes are Boolean vectors that can be stored at single-byte granularity. Let

| | |
|---|---|
| $D$ | dimensionality of each HV, |
| $C$ | number of material classes, |
| $F$ | number of distinct feature identities, |
| $L$ | number of quantisation levels per feature. |

Assuming 1byte per bit (packed storage can reduce this further), the total memory required is:

$$M_{\text{total}} \;=\; \frac{D}{1024}\big(C + F + L\big) \quad [\text{kB}]. \tag{17}$$

*Example.* With $D = 250$, $C = 5$, $F = 9$, and $L = 32$, Eq. (17) gives $M_{\text{total}} \approx 11.2\,\text{kB}$, of which 1.3kB are class prototypes, 2.2kB are feature HVs, and the remaining 8.0kB store quantisation-level HVs. Even a modest ARM Cortex-M MCU with 64kB of SRAM can therefore accommodate both the model and working buffers, leaving ample space for application code and sensor I/O.

# 5 Experimental Setup

To evaluate the performance of the proposed terrain classification system, a custom experimental setup was designed to simulate controlled surface interaction while capturing multimodal sensor data as shown in Figure 4. A custom 3D-printed wheel is driven on the wooden surface mounted on the rotating disk platform at the center, simulating the terrain interaction. Here, the current setup illustration uses wood as the surface material, while the platform is designed to allow easy replacement with other surfaces to assess classification across different terrain types. For the demonstration of our proposed system, we consider five types of surface textures for classification: aluminium metal, wood, cloth carpet, sand and fake-grass as illustrated in Figure 5, while this can be further extended to diverse kinds of surface textures.

## 5.1 Hardware Configuration

At the core of our system is the SparkFun Red Board Artemis ATP [6], powered by the Apollo3 SoC, which provides a low-power and high-performance micro-controller platform with an ARM Cortex-M4 Processor suitable for on-board data acquisition and inference. The board interfaces with both audio and vibration sensors and handles data collection and the HDC classification process. The sensor stack comprises:

(1) A **digital MEMS microphone** integrated into the Artemis board.
(2) A **piezoelectric film vibration sensor** (LDT0-28K [18] from SparkFun) with a breakout board.

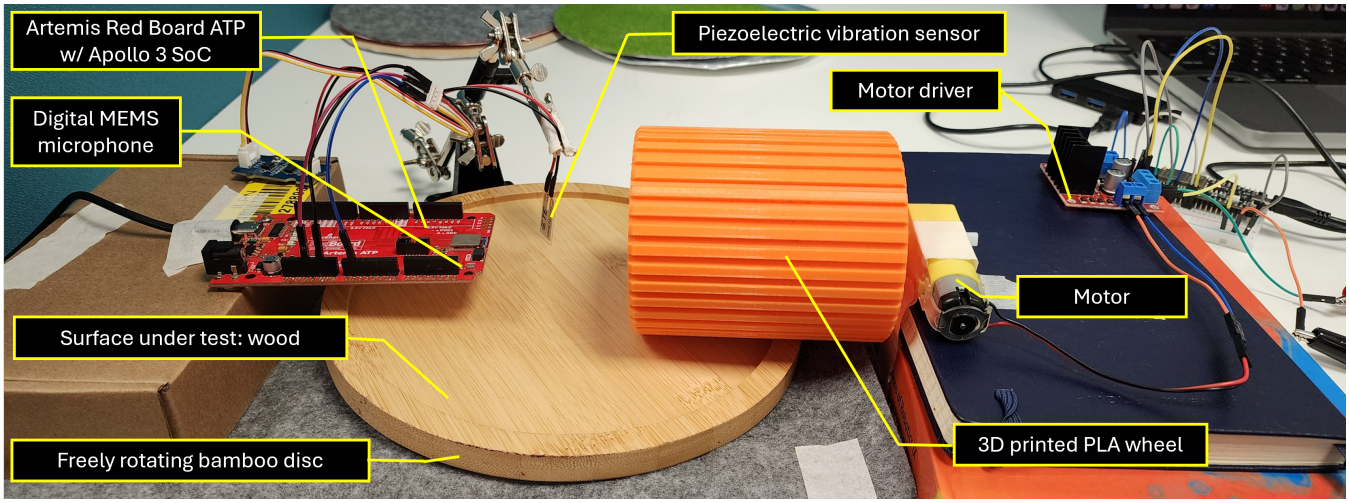



**Figure 4: Experimental setup for terrain classification. The setup includes a digital MEMS microphone and a piezoelectric vibration sensor interfaced with an Artemis Red Board ATP. A 3D-printed wheel driven by a DC motor interacts with a freely rotating bamboo disc simulating a wooden terrain. The entire system captures real-time data for acoustic and vibration-based classification.**

Both sensors are mounted close to the wheel–terrain contact point. The mic is mounted approximately 5-10 cm from this contact point, and the vibration sensor film is oriented perpendicular to the surface with its edge in contact with the terrain. The custom 3D-printed wheel is driven by a separate microcontroller and a DC motor via an L298N motor driver, creating controlled interactions with the terrain. The DC motor is powered via a regulated power supply and controlled using a microcontroller-based PWM signal. The motorized wheel operates at 8–9V, depending on the friction of the surface being tested.


**Figure 5: Five types of surface textures for classification: aluminium metal, wood, cloth carpet, sand and fake-grass.**

## 5.2 Data Collection Procedure

For collecting audio signals, data is sampled at 15.625 kHz, using a DMA-enabled I2S interface to transfer 4096 bytes every 262 ms with a 16-bit resolution. The pulse-density modulation (PDM) library is used to handle the data conversion to pulse-code modulation (PCM) audio format. For vibration data collection, data is sampled at the same rate 15.625 kHz, with a 10-bit resolution via the microcontroller's analog input interface.

Both audio and vibration data streams are transmitted over a serial interface to a host computer. A `Python` script is used to collect the data samples, convert them to required format (WAV for audio and CSV for vibration), and organize them for training. Each recording contains 5 seconds worth of data, and 35 recordings are collected for each surface material. Each sample includes a pair of synchronized files: one for audio and one for vibration data.

To ensure consistent and repeatable interactions across all surface types, we use the rotating bamboo platform equipped with circular discs of each terrain material. The motorized wheel is pressed against the freely rotating platform, generating consistent sound and vibration signals during movement for data collection. This design enables streamlined data collection and supports longer recordings when needed.

## 5.3 Dataset Preparation and Model Training

After collecting the data samples, we train an HDC classifier using the pipeline detailed in § 4. Training is performed off-device on a commodity computer that also hosts the entire dataset. A dedicated `Python` script built around the `librosa` library for feature extraction implements all HDC operations and learning from scratch. The dataset is partitioned into training and test splits in an 85:15 ratio. Once the model converges, the learned class hypervectors together with the modality-specific codebooks are exported as a `C` header

file, which is subsequently flashed to the microcontroller to enable on-board inference.

### 5.4 Testing on Live Data

Once trained, the model weights are loaded from a C header file that is compatible with the microcontroller's compiler. The inference Python code is converted to C code and ported to run on the Artemis Red Board ATP microcontroller to facilitate live inference along with live data collection. The experimental setup from the data collection procedure is used to test the live inference. The pipeline remains the same, except that the data is not streamed over to a computer, but retained and processed on the device. We report latency, memory and power values for the same in § 6. A live demo of this can be found in this demo video.

## 6 Evaluations

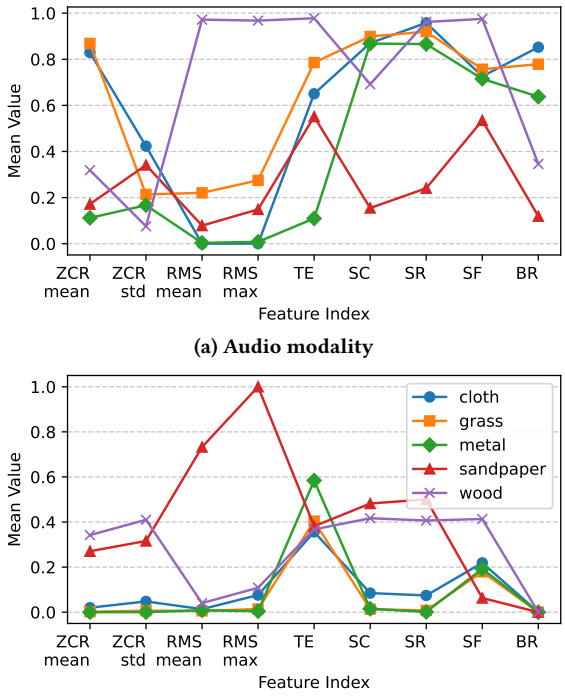
We evaluate the proposed HDC classifier along three axes: *(i)* the relative merit of audio and vibrational sensing, *(ii)* a head-to-head comparison with state-of-the-art (SoTA) machine- and deep-learning baselines, and *(iii)* the sensitivity of HDC to key design hyperparameters—hypervector dimensionality ($D$) and bit-quantisation resolution ($B$). All experiments use the dataset described earlier, split 85:15 for training/testing. Unless stated otherwise, results are averaged over five random initialisations. We also evaluate the performance and power of the HDC classifier deployed on the SparkFun Artemis RedBoard ATP microcontroller.

### 6.1 Acoustic *vs.* proprioceptive sensing

Figure 6 depicts the class-wise mean of the *normalised*[2] feature vectors extracted from the two modalities. The audio curves (Figure 6a) exhibit distinct, non-overlapping trajectories: for instance, *wood* dominates Features 3–4 while *metal* peaks sharply at Feature 6. In contrast, the vibration curves (Figure 6b) largely cluster near the baseline, with only *sandpaper* showing a pronounced peak at Feature 4. The separation in the acoustic domain therefore, provides richer discriminative information than proprioceptive vibration signals.

*Confusion–matrix analysis.* Figure 8 reveals how the class–wise behaviour mirrors the feature–space observations of Figure 6. With audio alone (Figure 8a) the classifier achieves a *perfect* diagonal: every one of the five held-out samples per class is correctly assigned, yielding 100% precision and recall for all classes. In stark contrast, the vibration–only model (Figure 6b) collapses largely onto the *metal* class: the first three materials are always mis-labelled as *metal*, and even *sandpaper*/*wood* suffer one misclassification each. This confirms that the vibrational features lack inter-class separability and explains the drop from 100% to 60% overall accuracy seen in Figure 7. Early-fusion of both modalities (Figure 8c) partially recovers lost performance—cloth, sandpaper, and wood now register true positives—yet residual confusion remains when acoustic evidence is weak (e.g. *wood* predicted as *cloth*). These results underscore two key insights: (i) acoustic cues dominate terrain discrimination, and

[2]Each feature is min–max scaled to $[0, 1]$ per modality before averaging.




**(a) Audio modality**

**(b) Vibration modality**

**Figure 6: Mean–normalised feature profiles for the five surface classes. Clear inter-class separation is visible in audio, whereas vibration features overlap substantially, explaining the lower stand-alone performance of vibration-only models.**

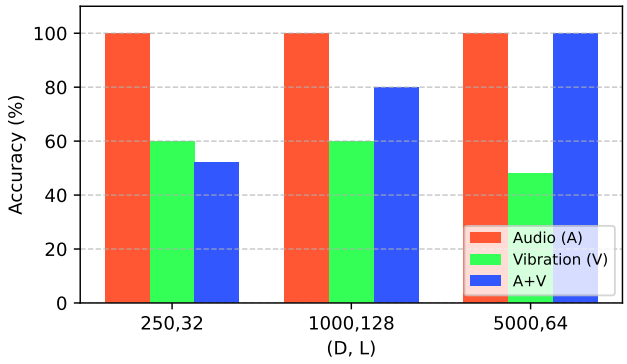
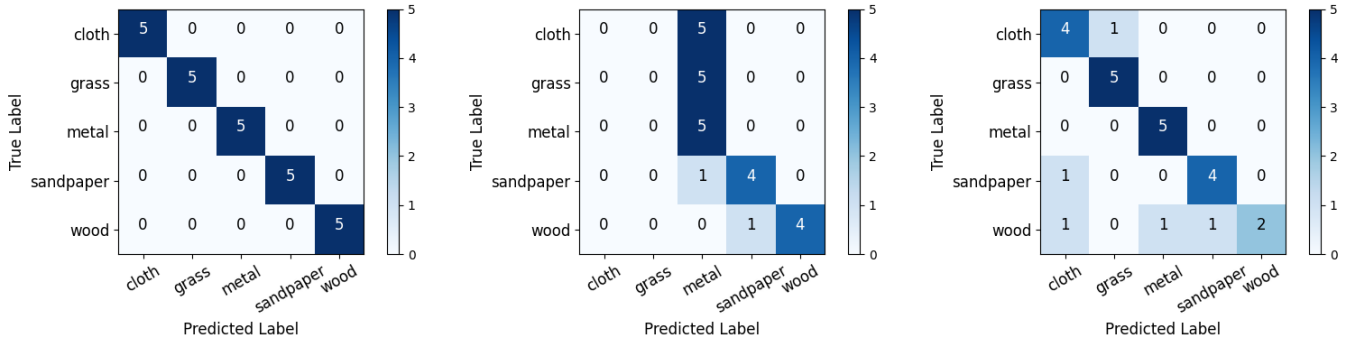



**Figure 7: Classification accuracy of HDC models trained on individual and fused modalities for three hypervector setups $(D, L) \in \{(250, 32), (1000, 128), (5000, 64)\}$. Audio provides a consistent upper bound; fusion benefits only once $D$ is large enough to preserve modality-specific structure.**

(ii) weak modalities can still be leveraged, but only once the hypervector dimensionality is high enough to preserve modality-specific structure (Section 6.3).

Figure 7 quantifies this observation. With a compact HD configuration $(D, L) = (250, 32)$, the audio-only model already reaches




(a) Audio modality (b) Vibration modality (c) Fusion (audio + vibration)

**Figure 8: Confusion matrices on the 15% test split for the HDC classifier ($D = 1000$ and $B = 256$) using (a) audio, (b) vibration, and (c) fused modalities. Each matrix shows absolute sample counts (5 per class). Perfect diagonal dominance in (a) contrasts sharply with the class-collapse in (b); fusion in (c) recovers much of the loss but still inherits some ambiguity from the weaker sensor.**

100% accuracy, while vibration stalls at 60%. Early sensor fusion (A+V) underperforms because the weak vibrational signal dilutes the strong acoustic representation. As dimensionality grows ($D = 5\,000$, $L = 64$) the fused model attains 100%, matching the audio-only ceiling and confirming that a sufficiently expressive HD space can absorb complementary—but noisy—modalities.

## 6.2 Comparison with state-of-the-art baselines

Figure 9 contrasts HDC with four widely used classifiers after log-scaling raw metrics to a 1–10 range (lower = better). HDC forms the smallest radar polygon, indicating the best overall trade-off:

- **Prediction error**. HDC achieves the minimum error envelope (100% accuracy), outperforming both classical (SVM, RandomForest) and deep (FF-NN, TinyCNN) counterparts.
- **Model footprint**. At barely 1.2kB[3] for $D = 250$, HDC is $\approx 3\times$ smaller than the lightest neural model.
- **Latency**. Profiling on a commodity i5 computer with single thread execution shows median inference latency of 0.1ms for HDC, versus 36.9ms for the TinyCNN baseline—an **369×** speed-up that translates directly into energy savings.
- **Training time**. Although SVM fits marginally faster on a desktop CPU, training is an *offline* step; once deployed, HDC's superior run-time characteristics dominate.

## 6.3 Impact of hypervector size and quantisation

The learning dynamics for varying $D$ and $B$ are plotted in Figure 10. Increasing $D$ accelerates convergence and raises the asymptotic accuracy up to about $D = 500$, beyond which gains plateau, confirming the theoretical saturation behaviour of HD classifiers. Bit resolution exhibits a different trend: all settings eventually converge to $\geq 97\%$, but coarse resolutions ($B=4$) require more epochs, while extremely fine ones ($B=4096$) slow down due to quantisation noise in early updates. In practice, $B \in [64, 256]$ offers an excellent speed-accuracy–memory trade-off.

[3]Note that we consider the boolean parameters to occupy 1 bit

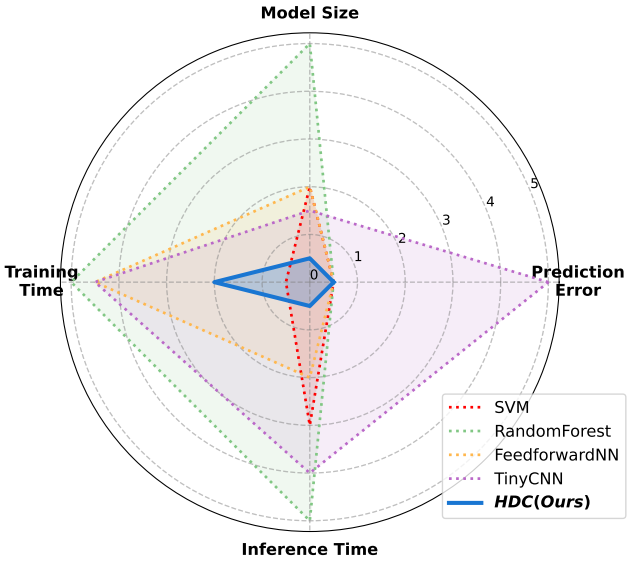
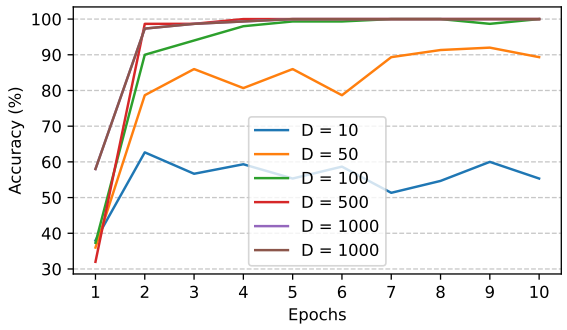


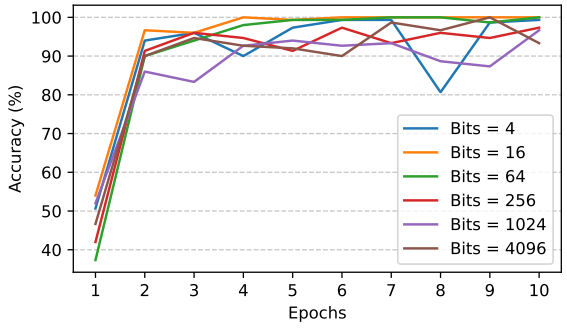

**Figure 9: Radar plot comparing HDC with SVM, RandomForest, Feedforward-NN and TinyCNN across four key metrics. Values are scaled from 1 (best) to 5 (worst) using a mix of linear and logarithmic transform to highlight relative differences. HDC offers the best overall balance, with especially strong gains in model size and inference time.**

## 6.4 Latency and Memory Measurement

To evaluate the runtime performance of the HDC inference pipeline on the SparkFun Artemis RedBoard ATP, we measured the latency from feature extraction to final classification and the memory consumption of the hyper-dimensional vectors. For latency measurement, the inbuilt millis() was used, where the function returns the number of milliseconds elapsed since the program started and is suitable for high-resolution timing. We inserted timing markers before and after the inference block to get the latency measurement. The reported latency value for onboard processing of feature




**(a) Effect of hypervector dimensionality ($D$)**




**(b) Effect of quantisation resolution ($B$)**

**Figure 10: Convergence of HDC classifiers over 10 training epochs. (a) Accuracy rises steeply with $D$ and saturates beyond $D \approx 500$. (b) Changing $B$ mainly alters convergence speed; final accuracy is nearly invariant, validating HDC's robustness to low-precision representations.**

extraction and HDC classification was 5.636s on average with 5 continuous runs of onboard HDC classification.

The memory measurement for hyper-dimensional vectors was also taken through the onboard inference code, which reported 34.69 KB accounting to 3.39% of the flash memory (1 MB) available on the Artemis RedBoard ATP micro-controller. The total program memory usage is 16% and global memory usage is 19% of available dynamic memory. The obtained latency and memory values indicate that our approach for using HDC for terrain classification is lightweight and memory efficient, making it suitable for deployment on resource-constrained embedded platforms.

## 6.5 Power Profiling

We conducted current measurements across all operational phases of the system. To do this, we used the Nordic PPK2 power profiler [17], and the `MEAS` pins on the Artemis ATP board to measure current draw. The profiling results, shown in Figure 11, highlight three key phases: initialization, sensing, and inference.

During the initialization phase, which lasts approximately 300 ms, the system consumes an average current of 1.83 mA as peripherals and communication interfaces are activated. In the subsequent sensing phase, lasting 5.03 seconds, audio and vibration data are sampled and buffered, drawing an average current of 1.01 mA. Finally, during the inference phase, the HDC-based terrain classification is performed on-device, consuming an average current of 914.82 $\mu$A over 5.6 seconds. This profile demonstrates the suitability of the system for deployment in energy-constrained robotic platforms. The idle phase shown at the end of the graph is not in sleep mode and is excluded from the current reporting. Note that the input voltage to the board was 5V and average current over the full cycle was 981.16 $\mu$A. Hence, the average power consumption is 4.9 mW.

## 6.6 Key takeaways

Across all experiments, HDC delivers *real-time*, *memory-light*, and *energy-efficient* surface classification:

(1) **Modality choice matters**. Rich acoustic cues dominate terrain discrimination, but with a sufficiently large HD space multimodal fusion can match the acoustic upper bound while adding redundancy against microphone failure.
(2) **HDC beats heavier models**. Despite its simplicity, HDC surpasses—or equals—SoTA classifiers while requiring an order of magnitude less compute and memory.
(3) **Parameter sweet-spot**. Choosing $D \approx 500$ and $B \in [64, 256]$ captures almost all attainable accuracy without bloating the codebook or slowing convergence.

# 7 Discussions

## Challenges with Vibration Sensor

While the piezoelectric vibration sensor is a critical component for capturing mechanical responses during terrain interaction, its integration into the experimental setup presented several challenges that had to be addressed to ensure reliable signal acquisition and system stability. Although piezoelectric sensors can detect a wide range of frequencies, their effective frequency response is often dependent on the material, form factor, and preloading conditions. It was observed that, to produce sensor output correlating to the forces experienced with the terrain interaction, the selected piezoelectric sensor film needs to bend sufficiently in both directions. This condition is not satisfied by terrain types with uniform texture like grass and cloth, which may have affected in the data collection process for vibration, resulting in most mispredictions in vibration only modality. Further exploration can be done with vibration modality by tuning the sensor using the potentiometer available with the breakout board. Also, the sensor was required to be firmly mounted against the surface to avoid loosening with mount point, due to the vibration experienced with the surface interaction, which can result in losing contact with the surface with continuous motion of the wheel.

## Improving Combined-Modality Accuracy

The overall accuracy of the audio–vibration model can be raised by re-balancing the data-fusion stage: either (i) down-weighting the contribution of vibration HVs, or (ii) retaining only the most discriminative vibration features before bundling. Both strategies suppress noisy or redundant vibration signatures, allowing the more informative audio cues to dominate the final HV and sharpen the classifier's decision boundary.

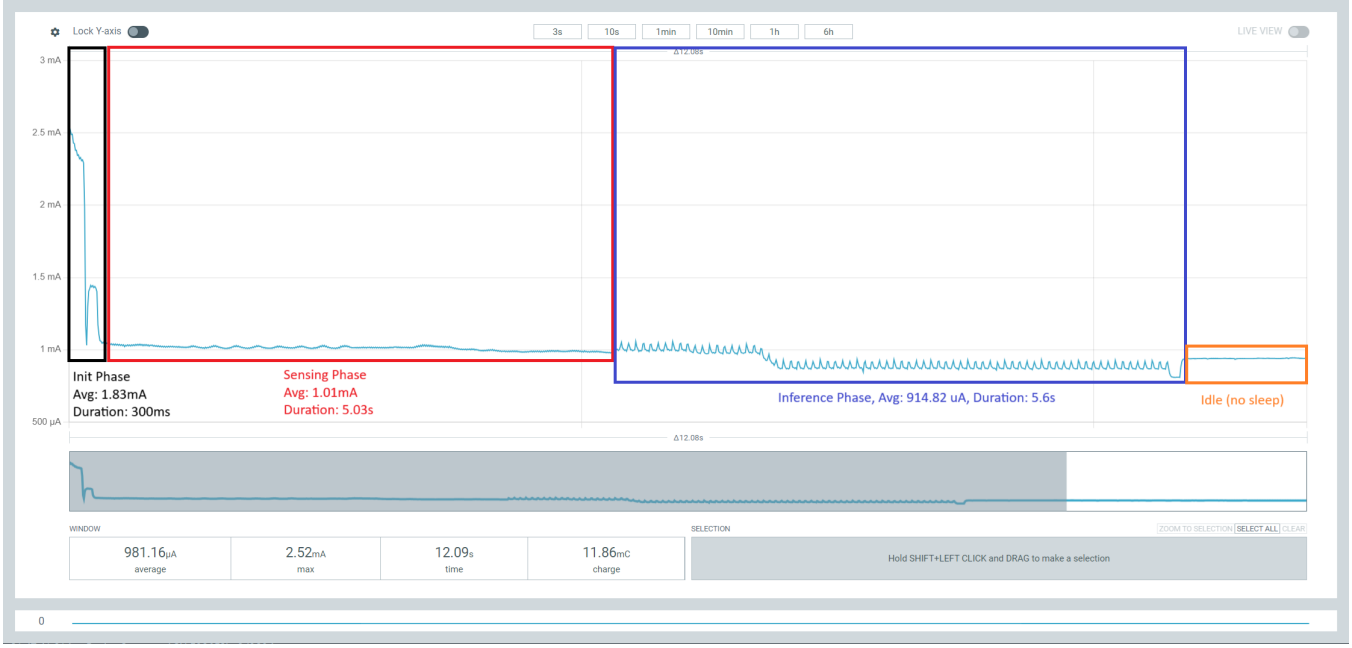



**Figure 11: Current draw over a cycle of initialization, sensing and inference.**

## Reducing Memory Footprint

In Eq. (17) we conservatively assumed one byte per bit; however, HD operations are defined in GF(2) and therefore can be applied with *bit-packing*. Storing eight HV bits per byte divides every term in the memory budget by 8, cutting the example footprint from ∼11.2 kB to just 1.4 kB without any loss of accuracy or additional compute overhead, because binding and bundling remain single-cycle XOR and popcount instructions on modern MCUs. Further savings are possible by (i) re-generating value-level HVs on-the-fly via deterministic permutations instead of storing all $L$ vectors. These optimisations can push the total RAM requirement well below 1 kB, making the classifier practical even for sub-$1 sensor nodes.

## Challenges with MCU Programming

While the SparkFun Artemis ATP board provided an excellent platform for low-power sensing and supposed a wide range or Arduino-compatible libraries, we encountered several issues during development. First, we observed that one some machines, the compilation and linking steps, especially for larger sketches with multiple headers were unusually slow. This introduced delays in iterating over the code and made quick testing cumbersome. More critically, we experienced frequent failures during the code flashing process. Uploaded would often stall or fail entirely, requiring repeated attempts or full resets. In some cases, the upload tool had to be switched to a different machine altogether to successfully program the board. The underlying cause of these issues remains unclear, but they may stem from the USB driver stack, timing issues in the bootloader, or differences in the host operating system environment. Despite these issues, once flashed, the board performed reliably during runtime. However, these flashing instabilities highlight a practical limitation in the development workflow that should be considered when scaling or deploying similar systems.

## Acknowledgments

We would like to thank Prof Ambuj Varshney and everyone else on the CS5272 teaching team for their support.

## References

[1] Unal Artan, Heshan Fernando, and Joshua A Marshall. 2021. Automatic material classification via proprioceptive sensing and wavelet analysis during excavation. In *2021 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*. IEEE, 612–617.

[2] Chengchao Bai, Jifeng Guo, and Hongxing Zheng. 2019. Three-dimensional vibration-based terrain classification for mobile robots. *IEEE Access*, PP, (May 2019), 1–1. DOI: 10.1109/ACCESS.2019.2916480.

[3] Amente Bekele. 2016. Cooley-tukey fft algorithms. *Technical Report*. https://people.scs.carleton.ca/~maheshwa/courses/5703COMP/16Fall/FFT_Report.pdf.

[4] Christopher A. Brooks and Karl Iagnemma. 2012. Self-supervised terrain classification for planetary surface exploration rovers. *Journal of Field Robotics*.

[5] Ayan Dutta and Prithviraj Dasgupta. 2017. Ensemble learning with weak classifiers for fast and reliable unknown terrain classification using mobile robots. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

[6] SparkFun Electronics. [n. d.] Sparkfun redboard artemis atp. https://www.sparkfun.com/sparkfun-redboard-artemis-atp.html. Accessed: 2025-04-24. ().

[7] Lulu Ge and Keshab K. Parhi. 2020. Classification using hyperdimensional computing: a review. In number 2. Vol. 20. (June 2020), 30–47. arXiv: 2004.11204. DOI: 10.1109/MCAS.2020.2988388.

[8] Eman Hassan, Zhuowen Zou, Hanning Chen, Mohsen Imani, Yahya Zweiri, Hani Saleh, and Baker Mohammad. 2024. Efficient event-based robotic grasping perception using hyperdimensional computing. *Internet of Things*, 26, 101207.

[9] Ignacio Heredia. 2019. Deep learning for audio classification. https://github.com/deephdc/audio-classification-tf. GitHub repository (archived 2024-05-02). (Sept. 2019).

[10] Alejandro Hernández-Cano, Namiko Matsumoto, Eric Ping, and Mohsen Imani. 2021. Onlinehd: robust, efficient, and single-pass online learning using hyperdimensional system. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 56–61. DOI: 10.23919/DATE51398.2021.9474107.

[11] Pentti Kanerva. 1997. Fully distributed representation. In *Proceedings of the Real World Computing Symposium (RWC)*, 358–365.

[12] Pentti Kanerva. 2009. Hyperdimensional computing: an introduction to computing in distributed representation with high-dimensional random vectors. In number 2. Vol. 1. (Jan. 2009), 139–159. DOI: 10.1007/s12559-009-9009-8.

[13] Chae Young Lee, Maxwell Fite, Tejus Rao, Sara Achour, Zerina Kapetanovic, et al. 2025. Hypercam: low-power onboard computer vision for iot cameras. *arXiv preprint arXiv:2501.10547*.

[14] Jacqueline Libby and Anthony J Stentz. 2012. Using sound to classify vehicle-terrain interactions in outdoor environments. In *2012 IEEE International Conference on Robotics and Automation*. IEEE, 3559–3566.

[15] MG Harinarayanan Nampoothiri, B Vinayakumar, Youhan Sunny, and Rahul Antony. 2021. Recent developments in terrain identification, classification, parameter estimation for the navigation of autonomous robots. *SN Applied Sciences*, 3, 1–14.

[16] YJ Ng, Matthew SK Yeo, QB Ng, Michael Budig, MA Viraj J Muthugala, SM Bhagya P Samarakoon, and RE Mohan. 2022. Application of an adapted fmea framework for robot-inclusivity of built environments. *Scientific Reports*, 12, 1, 3408.

[17] Nordic Semiconductor. [n. d.] Power profiler kit ii (ppk2). https://www.nordicsemi.com/Products/Development-hardware/Power-Profiler-Kit-2. Accessed: 2025-04-24. ().

[18] Measurement Specialties. [n. d.] Ldt0-028k piezoelectric vibration sensor. https://cdn.sparkfun.com/datasheets/Sensors/ForceFlex/LDT_Series.pdf. Accessed: 2025-04-24. ().

[19] Abraham Gebru Tesfay. 2015. Terrain type classification based on sound. (Oct. 2015).

[20] Anthony Thomas, Sanjoy Dasgupta, and Tajana Rosing. 2021. A theoretical perspective on hyperdimensional computing. In vol. 72. (Oct. 2021), 215–249. DOI: 10.1613/jair.1.12664.

[21] Abhinav Valada and Wolfram Burgard. 2017. Deep spatiotemporal models for robust proprioceptive terrain classification. *The International Journal of Robotics Research*, 36, 13-14, 1521–1539.

[22] Feng Xue, Yicong Chang, Tianxi Wang, Yu Zhou, and Anlong Ming. 2024. Indoor obstacle discovery on reflective ground via monocular camera. *International Journal of Computer Vision*, 132, 3, 987–1007.

[23] Feng Xue, Longteng Hu, Chen Yao, Zhengtao Liu, Zheng Zhu, and Zhenzhong Jia. 2022. Sound-based terrain classification for multi-modal wheel-leg robots. In *2022 international conference on advanced robotics and mechatronics (ICARM)*. IEEE, 174–179.

[24] Sanggeon Yun, Ryozo Masukawa, Hanning Chen, SungHeon Jeong, Wenjun Huang, Arghavan Rezvani, Minhyoung Na, Yoshiki Yamaguchi, and Mohsen Imani. 2025. Hyperdimensional intelligent sensing for efficient real-time audio processing on extreme edge. In Early Access. (Feb. 2025). arXiv: 2502.10718. DOI: 10.48550/arXiv.2502.10718.