School of Computation, Information and Technology
TECHNISCHE UNIVERSITÄT MÜNCHEN

Forschungspraxis

Invasive Neural Data Analysis and Spike Sorting in Locusts

Author: Chutong Ren
Supervisor: Hu Peng
30.10.2024

1 Introduction

Microelectrode arrays (MEAs) have not only the clinical potential for treating diseases, such as obesity and diabetes[1], but also pave the way to neuroscience research by enabling precise stimulation and recording of neural signals[2].

Traditional 2D MEAs are flat and limited to surface contact, which restricts their ability to capture more complex, three-dimensional neural activities. The Utah array, a widely used 3D MEA, is effective in many clinical and research settings[3]. However, it is too large for small animals, like mice, and its performance can decrease over time due to foreign body responses.

To address these limitations, a flexible 3D MEA with dimensions of 0.9 x 0.9 cm^2 was designed for recording neural data in small organisms, such as the locust's metathoracic ganglion in the chest area.

In this report, we aim to verify the functionality of the flexible 3D MEA by evaluating its ability to capture neural signals from the locust's nervous system. Data analysis was conducted using Python, Offline Sorter4, Final Cut Pro X, and Photoshop.

2 Materials and Methods

2.1 Construction and Capabilities of the Flexible 3D MEA

The flexible 3D MEA consists of three parts: the electrode area, feedline area, and contact pad, as shown in Fig.1. Parylene was used as the substrate, followed by inkjet-printing of 3D shanks with gold and fabrication of conductive traces using photolithography. Next, another passivation layer was added with parylene on top. Finally, laser ablation was used to expose the tips of the shanks to make them conductive.

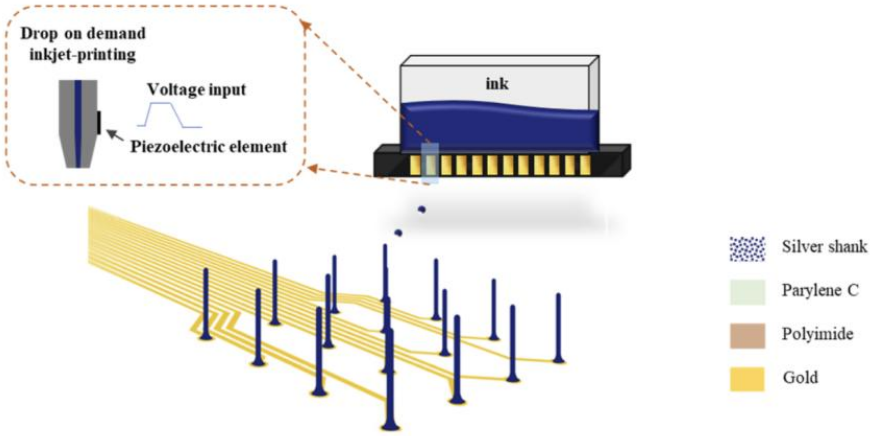

Fig. 1. The existing 3D MEA.

Tabel 1. Parameters and capabilities of the current MEA

Parameters	Capabilities
Pitch(density)	60-80μm
Electrode(shank) diameter	40-60μm
Shank count	16
Substrate material	Parylene
Feedline width	20μm
Gap between each feedline	20μm

2.2 Experimental subjects

We chose locusts mainly due to their small size, with a metathoracic ganglion around 1 x 1 cm^2, closely matching the dimensions of our MEA. This compatibility allows for precise targeting of neural regions. Additionally, the relatively simple nervous system of locusts enables visually observable responses to electrical stimulation.

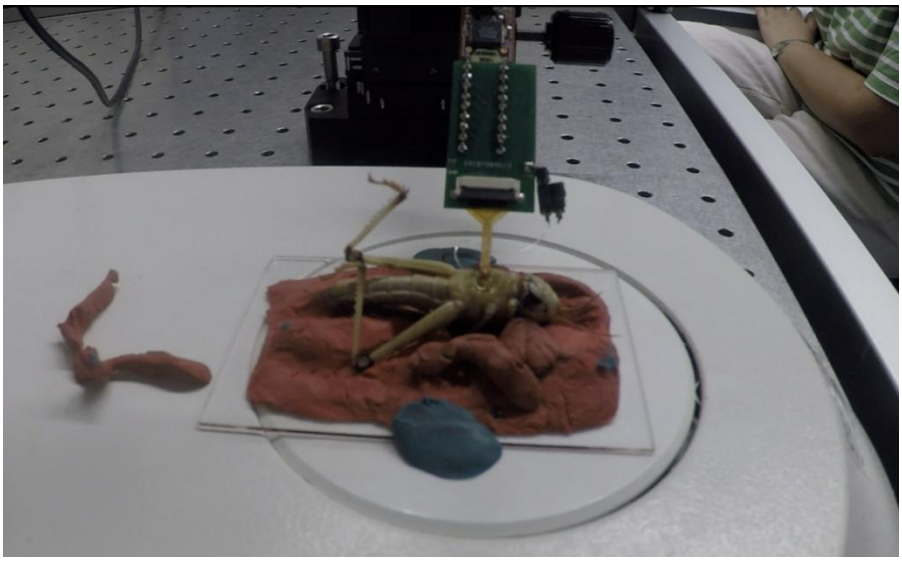

Fig. 2. Experimental scheme. The locust's body is secured with clay, and the thoracic cavity is dissected to expose the ganglion. The flexible 3D MEA is inserted into the neural region, with electrical currents applied to different electrodes.

2.3 Datasets

The invasive neural data collected has a sampling frequency of 30,000 Hz, a duration of 60 seconds, and includes 14 channels (with 2 reference channels). The dataset attributes include "amplifier_data", "frequency_parameters", "t", etc. Each dataset file is named in the format "20240730_Hu_device2Au_recording1_video_240730_193i48.rhs", where i indicates the dataset index (ranging from 1 to 4). In total, four datasets were collected.

Simultaneously, locust movements were recorded using a GoPro camera. The recorded videos are as follows:

Table 2. Parameters of experimental recorded videos

Video Name	Stimulated Electrode	Movement
GOPR7384	only test before experiments	Still
GOPR7385		Still
GOPR7386	E1	Outside leg
GOPR7387	E2	Inside leg
GOPR7388	E1, E5 E2 E3, E4, E6-E9	Subtle movement Inside leg Still
GOPR7389	E9-E16	Still

2.4 Data Processing

2.4.1 Data Reading

The data, collected using Intan Technologies' equipment, is in .rhs format. To read these data in Python, we explored two primary methods. The first method utilizes the SpikeInterface library, which can be installed with the command “pip install spikeinterface”. By importing “read_intan” from “spikeinterface.extractors”, we can load the data, specifying the “file_path” to the .rhs file and setting “stream_id='0'” to access amplifier data.

The second method involves using the "Python RHS file reader" provided on the official Intan website. However, running this code may lead to several common issues. An “ImportError: No module named intanutil.header” may occur due to the modular structure of the code, where “main.py” cannot locate functions in other modules. This issue can be resolved by directly copying the required functions into main.py. Another potential error is “ValueError: more than two arguments were supplied”, which likely originates from the margins function that may not accept both keyword and positional arguments simultaneously. This can be fixed by adjusting the code to use “margins(0, 0)” in load_intan_rhs_format.py. Additionally, an ImportError for missing modules (e.g., “find_peaks”) can occur even if “scipy.signal” package is installed, often due to conflicts between Python environments. This issue can be resolved by creating a new virtual environment specifically for this project and reinstalling the necessary packages. Once these issues are addressed, the program can be executed in the terminal with the command “python main.py <datapath>”, such as “python main.py data/20240730_Hu_device2Au_recording1_video_240730_193348.rhs”.

2.4.2 Data Analysis

To verify that the flexible 3D MEA can accurately capture neural activity in locusts, we focused on detecting action potentials (APs), as they are a key indicator of neuronal firing. An AP is a rapid, temporary change in a neuron’s membrane potential that enables signal transmission. This process begins when the neuron reaches a threshold potential, leading to the opening of voltage-gated sodium channels and an influx of Na ions. This results in a sharp rise in membrane potential, known as depolarization, which peaks around +30 mV. Shortly afterward, sodium channels close, and potassium channels open, allowing K ions to exit the cell, returning the membrane potential to its resting state through repolarization. Sometimes, this downward shift overshoots, causing hyperpolarization, before the membrane stabilizes around -70 mV as the sodium-potassium pump restores ion balance. This cycle produces the AP’s distinctive rise-and-fall pattern. This rise-and-fall cycle of an AP is a characteristic feature of neural signals, making it a reliable indicator for evaluating MEA performance.

To clearly identify AP signals, we applied a 300-5000 Hz bandpass filter to isolate relevant frequencies, effectively reducing background noise and preserving neural signal features. After filtering, we used the “find_peaks” function from “scipy.signal” package to detect spikes,

marking each action potential. The presence of distinct AP spikes is essential to validate the MEA's ability to capture locust neural activity.

2.5 Spike Sorting

So far, we have demonstrated that the flexible 3D MEA works, as significant action potentials (APs) have been observed after data spikes detection. To take this further, we aim to identify which neuronal units the spikes originate from. Spike sorting groups detected spikes into clusters, each potentially representing a unique neuronal unit. Spike sorting typically involves four steps[4]: first, bandpass filtering; second, spike detection; third, feature extraction; and finally, clustering, as shown in Fig. 3. Spike Sorting is considered unsupervised learning since we don't have correct labels for the spikes and don't know in advance which spikes belong to which neurons. Therefore, we cannot use classification algorithms such as SVM.

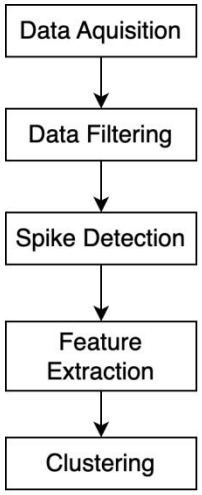

Fig. 3. Flow Chart of Spike Sorting

An analogy to spike sorting is Blind Signal Separation. Given in a noisy room with multiple voices, different sound sources overlap, but each microphone records a mixture. Suppose there are four people (S_1, S_2, S_3, S_4, where S stands for Sources) and four recorders. Recorder 1, for example, can be represented as a combination of the sources, specifically as $a_{11}S_1 + a_{12}S_2 + a_{13}S_3 + a_{14}S_4$. Similarly, Recorder 2 can be represented as $a_{21}S_1 + a_{22}S_2 + a_{23}S_3 + a_{24}S_4$. In general, we can write:

$$\text{recorder}(i) = a_{i1}S_1 + a_{i2}S_2 + a_{i3}S_3 + \ldots + a_{ij}S_j,$$

Here, i represents the number of recorders, and j represents the number of voice signals, or in other words, the number of voice sources or individuals. In analogy to neural activity, different neural units can be considered as the sources S in this equation. The weights can be regarded as reflecting the influence of different neural units.

2.5.1 The Combination of DWT and K-means

In Section 2.4, the first and second steps of spike sorting have already been completed. The third step in spike sorting processes involves feature extraction for dimensionality reduction, translating data from a high-dimensional space (2ms * 30000 sample points/s = 60 datapoints per spike in our case) to a few essential features. Ideally, these features should best separate the spike clusters. Commonly considered features include peak amplitude, peak-to-peak amplitude, width, and energy (the square of the signal). However, research has shown that such simple features may not be optimal for distinguishing spike shapes in general[5]. Some algorithms can extract the most effective features for clustering, for example by using a linear combination of channels with weighted contributions for each.

The most widely used feature extraction method is principal component analysis (PCA), which reduces data to the first two or three principal components, often capturing over 80% of the signal energy. While PCA captures the largest variances, these dimensions may not always best distinguish clusters, prompting us to explore Discrete Wavelet Transform (DWT) as an alternative. Unlike PCA, DWT preserves both time and frequency information, allowing for the preservation of localized data characteristics. DWT decomposes the original waveform layer by layer, using two wavelet functions: one captures low-frequency (general) components, and the other extracts high-frequency (detail) information. This layered decomposition keeps track of when certain frequencies occur, providing a time-frequency representation of the signal with high resolution in both domains[6].

The fourth and final step of spike sorting is to group spikes with similar features into clusters, representing different neural units. K-means achieves this by minimizing the distance between each data point and the center of its assigned cluster. It begins by randomly selecting k initial cluster centers, then iteratively assigns each spike to the nearest center, recalculates the cluster centers as the mean of the assigned points, and repeats until the clusters stabilize.

Then we realized DWT and K-means by writing four functions in Python. The spike data, denoted as spike_data, has dimensions of (n_spikes, n_samples), where n_spikes is the number of spikes and n_samples is 60 in our case. We applied the DWT to each spike, using the Daubechies 16 wavelet (db16) and decomposing the signal into 8 levels. The “pywt.wavedec()” function was used to obtain the wavelet coefficients for each spike, which were then concatenated into a single vector to represent the spike. This resulted in a feature matrix where each spike is represented by a long vector of wavelet coefficients. After extracting the features, we performed feature selection using the Kolmogorov-Smirnov (KS) test. For each feature, the KS test was applied to check if it follows a normal distribution. If the p-value from the test was less than the significance level alpha = 0.05 (default), the feature was considered significantly non-normal and was retained. The selected features, which form a 2D array with the shape (n_spikes, n_selected_features), were then used as input for the K-means clustering algorithm. In addition to the selected features, the number of clusters must also be specified as input. The

KMeans algorithm then grouped the spikes into clusters based on their wavelet features, and the output of this process was the cluster labels for each spike.

This method is popular for its simplicity and efficiency. However, a key limitation of K-means is the need to predefine the cluster count, which is unknown in unsupervised spike sorting, prompting the use of more adaptive clustering methods. To address this, we aim to use algorithms that can automatically determine the optimal number of clusters. State-of-the-art clustering methods are available in Offline Sorter 4.0, such as valley sampling and scanning K-means, offer this capability. Therefore, we use Offline Sorter 4.0 to improve the accuracy of spike sorting by dynamically adapting to the data.

2.5.2 Offline Sorter 4.0

The process of using this software is similar to the spike sorting process. To start, import the data file by navigating to the File tab and select “NeuroExplorer File (.nex file).” On the official Intan Technology website, under the RHD/RHS File Utilities section, there is a tool called the NEX file converter, which can convert .rhd or .rhs data files into NeuroExplorer format. This format can then be imported into Plexon’s Offline Sorter 4.0.

Then, perform the filtering step by clicking on the “Waveforms” tab, then “Filter Continuous Data,” and choosing a Butterworth filter with a 300 Hz cutoff. Next, select a single channel in the source panel, and a raw data plot will appear in the Timeline panel.

Next, we perform spike detection. Move the red line on the Neural Recording plot (where the x-axis represents time and the y-axis represents voltage) to adjust the threshold. We can also set the length of the spikes and define their start and end points. In the Timeline Panel’s left section, select “Detect Waveforms,” click “Options,” and set the Waveform Length to 2000, Prethreshold Period to 1000, and Dead Time to 2000. Then, in the Waveform tab, choose “Detect,” select “Use Current Thresholds,” and click “Detect.” This will select all spikes that exceed the threshold, which will appear in the Waveforms Panel.

In the 2D Clusters Panel, you can view scatter plots of the PCA feature extraction results. The x-axis represents the first principal component, and the y-axis represents the second principal component.

Lastly, go to the Sort tab, select "Change Sort Method," and choose from 12 available methods: Box, Lines, Bands, K-Means, Standard E-M, Valley Seeking, T-Distribution E-M, Scanning K-Means, Scanning Standard E-M, Scanning Valley Seeking, and Valley Seeking T-Distribution E-M. Valley Seeking has the best performance. The principle of it is to identify clusters by locating "valleys" in the data density landscape. The algorithm moves data points towards regions of lower density between high-density clusters, effectively grouping points in separate clusters

where the density “peaks” are separated by “valleys.” This approach doesn’t require pre-defining the number of clusters and adapts to the natural structure of the data.

2.6 Data Visualization

To observe the angle change of the locust leg’s movement more clearly and directly, Photoshop (PS) was used. First, two frames were exported from the video using Final Cut Pro X (FCPX). For video 387, where the outside leg moved slightly, the frames at 27:01 (before) and 27:16 (after) were chosen. For video 386, where the inside leg moved slightly, the frames at 46:17 (before) and 47:05 (after) were selected.

Click the “Share” icon in the top-right corner, then select “Save Current Frame” to export the current frame in JPG format. Next, import these two frames into PS and use the pen tool to outline the shape of the leg in the after frame. In the “Path” panel, choose “Load path as a selection” to convert the shape into a selection. Then, create a new layer in the “Layers” panel, which will represent the shape of the leg after the movement. This allows for a more apparent comparison of the angle change. Lastly, go to Filter - Blur - Motion Blur to give the after frame a sense of movement.

3 Results

Fig. 4 shows neural recordings from a 14-channel setup over a period of 60 seconds, with each channel labeled from CH1 to CH14. The x-axis represents time in seconds, while the y-axis shows voltage in microvolts (μV). Each channel displays the raw neural signals captured during the experiment.

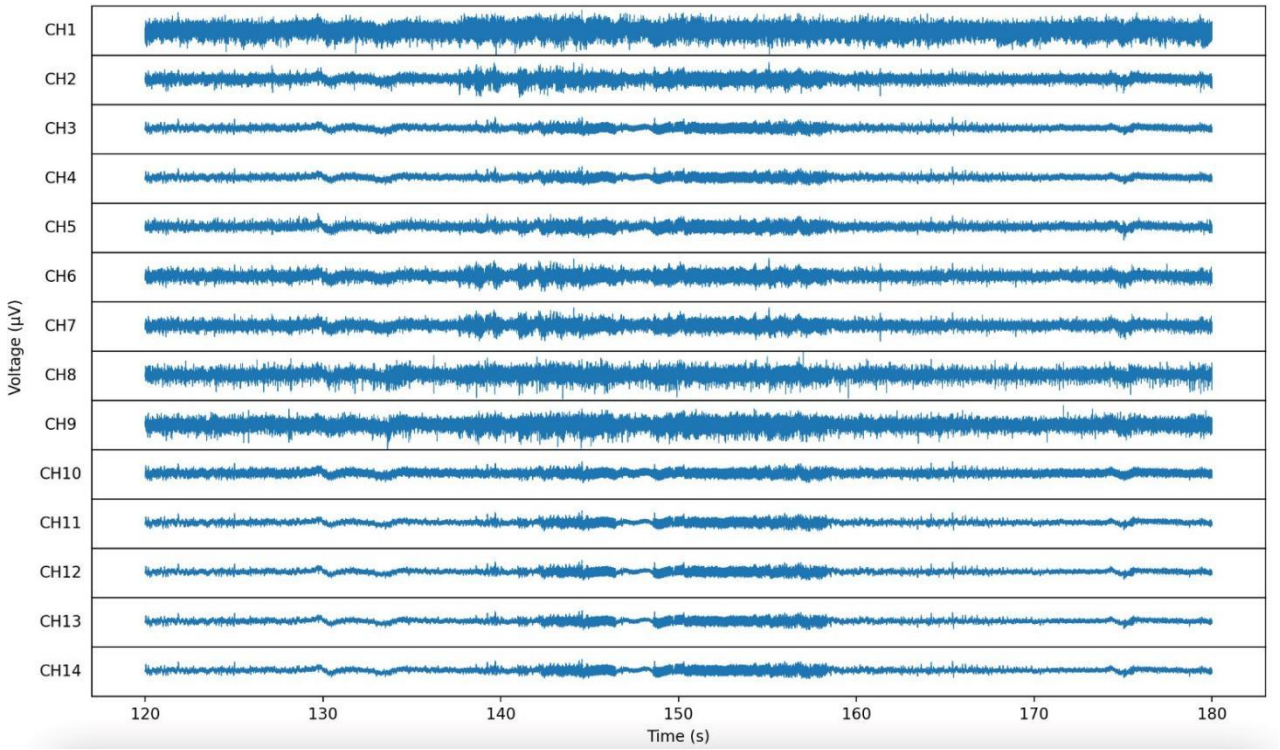

Fig. 4. Raw Data of Dataset 3, with the x-axis representing time from 120s to 180s and the y-axis representing 14 channels.

Fig. 5 shows neural recordings from 14 channels after applying a second-order Butterworth bandpass filter (300-5000 Hz). Channels CH1, CH2, CH5, CH6, CH7, CH8, and CH9 display clear, consistent neural activity, indicating effective filtering and good signal quality. Other channels show minimal activity or noise, suggesting weaker or less reliable signals.

Then we selected the filtered data of CH8 and zoomed in from 159s to 179s, where many obvious spikes can be seen, as shown in the left figure of Fig. 6. When the threshold was set to 50 μV, 3115 spikes were selected from the filtered data of CH8, as shown in the right figure of Fig. 6.

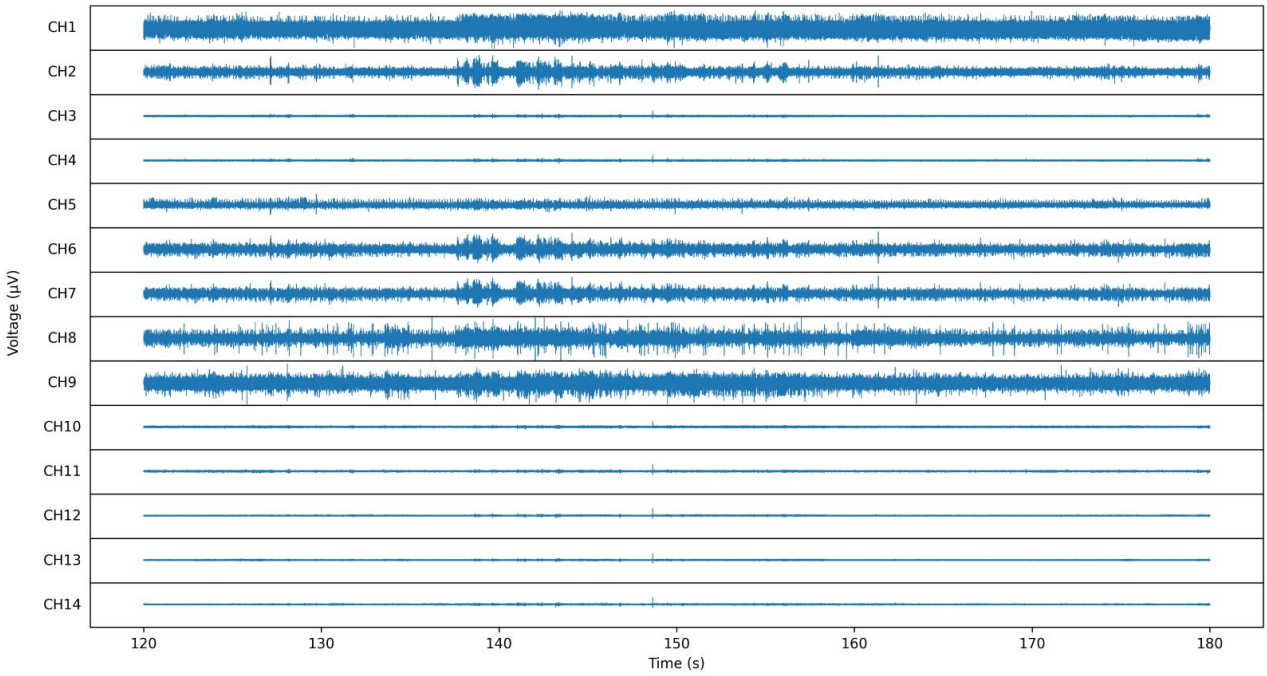

Fig. 5. Filtered Data after 2nd-order Butterworth Filter with a 300-5000 Hz Bandpass. Channels 1, 2, 5, 6, 7, 8, and 9 perform well.

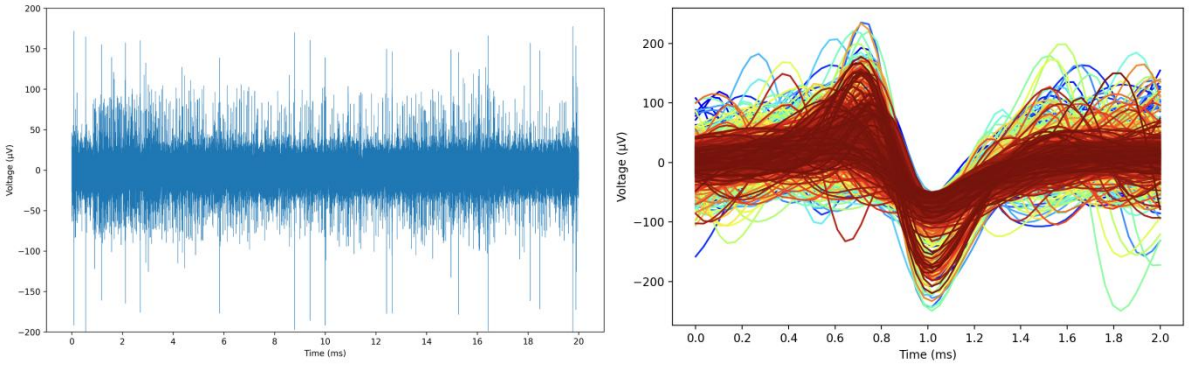

Fig. 6. For Channel 8, Left: Zoomed-in 20s data extracted from 159s to 179s. Right: All spikes selected with a 50 mV threshold.

The number of selected features is 304. For simplicity, Fig. 7(a) displays only the relationship between the first two features. These two features, as shown, exhibit a non-linear but distinguishable separation, which aids in clustering. When the number of clusters is set to 2, the clustering outcome is presented in Fig. 7(b), with blue points representing spikes from one neural unit and red points indicating spikes generated from another neural unit. By averaging the spikes within each cluster, we obtained the characteristic waveform shapes for the two units, as shown in Fig. 7(c) and (d) respectively.

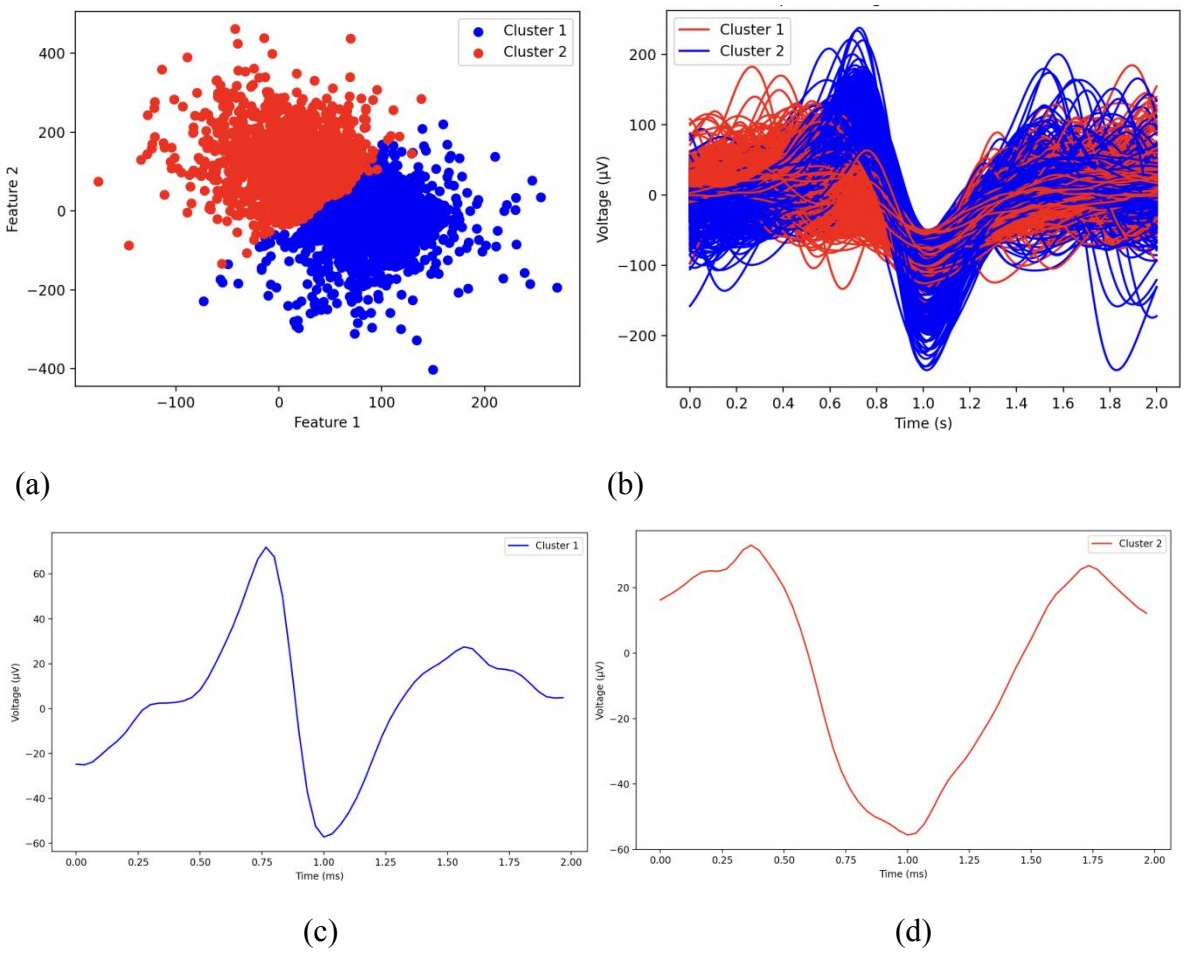

Fig. 7. DWT + KMeans Clustering Using ‘db16’ Wavelet, Level 4. The figure illustrates the feature extraction and clustering results from applying DWT and K-means to spike data.

In Offline Sorter 4.0, PCA is the built-in feature extraction method. Figure 8(a) shows the relationship between the first and second principal components. After trying various clustering methods provided by Offline Sorter, we observed that most methods, except for T-Distribution E-M and Scanning K-Means, clustered two neural units. T-Distribution E-M resulted in five clusters, likely due to the reduced overlap of extracted features, revealing distinct groups of spikes. Scanning K-Means identified three clusters, although units 'b' and 'c' appeared too similar to distinguish clearly. The two waveforms of clustered spikes are displayed in Fig. 8(c), where solid lines represent the mean waveform of each spike cluster. The dashed lines indicate the average trace, providing a general shape of the spike waveform for each unit across all recorded spikes.

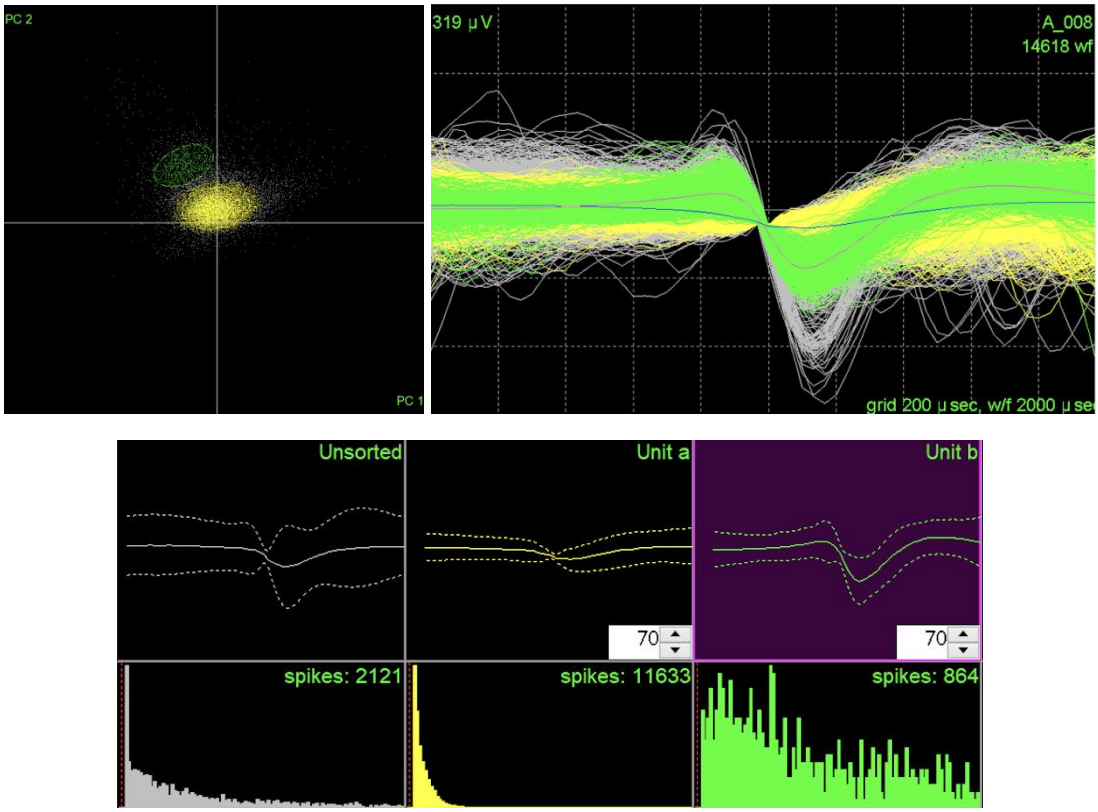

Fig. 8. Clustering results using Offline Sorter 4.0. (a) Visualizes the relationship between the first and second principal components derived from PCA. (b) Displays the spike waveforms of two identified units. (c) Illustrates the unsorted spikes, and (d) shows spikes in each unit with the count displayed below.

Obvious leg movements are observed in response to electrical stimulation. In video 387, there is noticeable movement in the inner small leg, while video 386 shows distinct movement in the outer large leg.

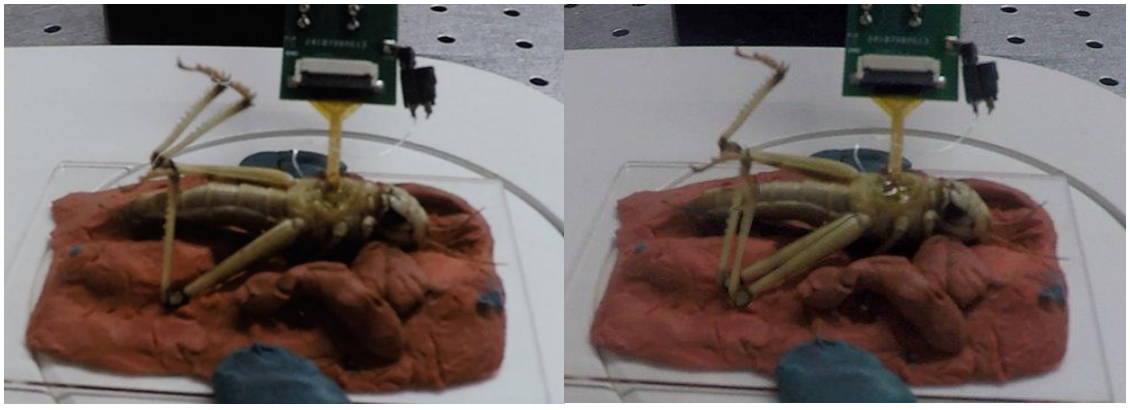

Fig. 9. Leg mvement observed in response to electrical stimulation.

4 Discussions

The flexible 3D MEA demonstrated its effectiveness, successfully capturing neural activity, which suggests that the recorded signals originate from two distinct neural units after spike sorting analysis.

In future work, we plan to explore how locust leg neurons respond to various types and intensities of electrical stimulation. Specifically, we will examine responses to step functions, where the input changes abruptly, and ramp functions, where the input increases gradually. This analysis aims to establish a functional relationship between the stimulation current and the resulting leg movement angle, offering further insights into how different stimulation patterns influence neuronal activity in locusts.

Code Availability

The code for this project can be found under:

https://github.com/chutongren/FP

References

[1] Hiendlmeier L, Zurita F, Vogel J, et al. 4D-Printed Soft and Stretchable Self-Folding Cuff Electrodes for Small-Nerve Interfacing[J]. Advanced Materials, 2023, 35(12): 2210206.

[2] Saleh M S, Ritchie S M, Nicholas M A, et al. CMU Array: A 3D nanoprinted, fully customizable high-density microelectrode array platform[J]. Science Advances, 2022, 8(40): eabj4853.

[3] Brown M A, Zappitelli K M, Singh L, et al. Direct laser writing of 3D electrodes on flexible substrates[J]. Nature Communications, 2023, 14(1): 3610.

[4] R ey H G, Pedreira C, Quiroga R Q. Past, present and future of spike sorting techniques[J]. Brain research bulletin, 2015, 119: 106-117.

[5] Quiroga R Q, Nadasdy Z, Ben-Shaul Y. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering[J]. Neural computation, 2004, 16(8): 1661-1687.)

[6] Letelier J C, Weber P P. Spike sorting based on discrete wavelet transform coefficients[J]. Journal of neuroscience methods, 2000, 101(2): 93-106.