The Neuromorphic Control MATLAB toolbox

Christian Fernandez Lorden[1], Fulvio Forni[3], Guillaume Drion[1], Pierre Sacré[1], Alessio Franci[1,2]
[1]Montefiore Institute, University of Liège, Belgium [2] WEL Research Institute, Wavre, Belgium
[3] Control Research Group, University of Cambridge, UK
email : Christian.FernandezLorden@uliege.be, ff286@cam.ac.uk
gdrion@uliege.be, p.sacre@uliege.be, afranci@uliege.be

1 Introduction

In the realm of control systems and robotics, the quest for intelligent and adaptive controllers has led researchers to draw inspiration from the intricate workings of the human brain. This pursuit has prompted a departure from traditional control concepts [1]. Traditional control, often expressed in terms of continuous-time signals, contrasts with the largely event-based nature of neurons.

To be able to use these concepts and models in a control workflow, we present a toolbox that provides blocks based on the work of Ribar *et al.* [2] to quickly create neuronal circuits that can be inserted in a control loop. This allows modeling, design, and testing of neuromorphic controllers to be far quicker and accessible to a wider audience.

2 Computing with events

For neuromorphic control to make sense and to better understand how to use this toolbox, it is necessary to understand the basic paradigm of neural computation. Neurons communicate with each other by sending spikes, or events, to each other [3].

Therefore, while technically simulating continuous-time signals, this toolbox is designed to work with spiking events in mind (see Figure 1 top graph). The intricacy of working in an event-based paradigm is that blocks should be designed to receive semantic 1-bit events from other blocks and generate 1-bit events to other blocks, while permitting local continuous signals to exist.

3 The Toolbox

The toolbox currently defines only six blocks, two of which are the main blocks that define the other blocks. These two basic blocks are the low-pass filter and the saturation block. Using those blocks, four other blocks are defined: the neuron, the synapses with facilitation and depression, and what we call the neuromodulatory synapse. These building blocks are meant to be combined to create more complex computing blocks that then adhere to the event-based paradigm of Section 2. The toolbox is available on the Mathworks File Exchange [4].

4 Applications

Using these basic computational blocks, it is possible to realize a full control loop around defined sensors and actuators. This means defining event-based sensory transducers, event-based actuator transducers, and event-based computation blocks.

A simple and useful example of a computational block that you can create is a zero-crossing detector. This block comes up naturally in an event-based computing framework, since semantic events are often linked to a signal crossing a threshold.

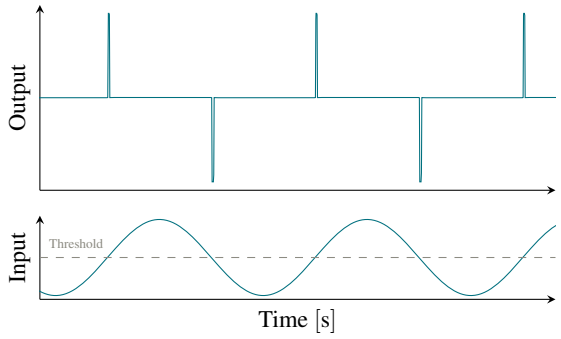

Figure 1: Zero-crossing detector output when subjected to an alternating input signal.

Figure 1 displays the behavior of an event-based zero-crossing detector built using the toolbox. This detector distinguishes between a positive and a negative crossing and communicates it by the polarity of the event.

References

[1] S. P. DeWeerth *et al.*, "A simple neuron servo", 1991.

[2] L. Ribar *et al.*, "Neuromodulation of neuromorphic circuits", 2019.

[3] G. Piccinini *et al.*, "Neural computation and the computational theory of cognition", 2013.

[4] C. F. Lorden, *Neuromorphic toolbox for control*, https://www.mathworks.com/matlabcentral/fileexchange/178514-neuromorphic-toolbox-for-control, [Online; accessed January 6, 2025], 2025.