UNIVERSIDAD AUTÓNOMA DE MADRID

ESCUELA POLITÉCNICA SUPERIOR

TRABAJO FIN DE MÁSTER

Hybrid experimental and model approaches for the caracterization of *Lymnaea stagnalis* neural activity
Part II:
Design and implementation of hybrid circuits in *Lymnaea stagnalis*

Máster Universitario en Ingeniería Informática

Autor: Alicia Garrido Peña

Tutor: Pablo Varona Martínez
Departamento de Ingeniería Informática.

Diciembre, 2019

Abstract

Hybrid circuits bring together living and model neural activity, adapting each other into one unique circuit. This technique allows advanced studies of neural dynamics and the implementation of novel experimental protocols. In this project, a hybrid experiment has been designed to test the interaction of dynamic synapses and intrinsic neural properties such as subthreshold oscillations in shaping non-trivial responses to simple spike trains. For this purpose, a conductance-based model with subthreshold oscillations has been studied with this type of stimulation in order to run a prove-of-concept experiment to map input/output relationships. Moreover, it has been implemented in a real-time software tool, RTXI, to induce subthreshold oscillations into a quiescent living neuron and to perform the analysis in living neurons. This report also discusses additional validation experiments on this approach by using RTXI implementations in different scenarios.

Keywords— Hybrid circuits, Lymnaea Stagnalis, neural sequencies, non-trivial input/output neuronal responses, subthreshold oscillations, RTXI.

Resumen

Los circuitos híbridos aunan actividad de neuronas vivas y modelos, adaptando ambos en un único circuito. Esta técnica permite realizar estudios avanzados de dinámicas neuronales, así como la implementación de protocolos experimentales novedosos. En este proyecto, se ha diseñado un experimento híbrido con el fin de probar la interacción de sinápsis dinámicas y propiedades neuronales intrínsecas como las oscilaciones subumbrales para construir respuestas no triviales a secuencias de potenciales de acción. Con este fin, se ha estudiado un modelo basado en conductancias con oscilaciones subumbrales con este tipo de estimulación, con el objetivo de llevar a cabo un experimento de prueba de concepto para mapear relaciones de entrada/salida. Además, se ha implementado en una herramienta de tiempo real, RTXI, para inducir oscilaciones subumbrales en una neurona viva con actividad silenciosa y poder realizar el análisis en neuronas vivas. Este documento abarca experimentos de validación adicionales en este enfoque usando implementaciones de RTXI en diferentes escenarios.

Palabras clave— Circuitos híbridos, Lymnaea Stagnalis, secuencias neuronales, respuestas input/output no triviales, oscilaciones subumbrales, RTXI.

Acknowledgements

First, I would like to thank this project to GNB. To Pablo, for introducing me the Neurocomputing world, as well as all the hours spent and the trust he has put on me. Also, to Irene and Rafi, for all the hours doing experiments and for having enough patience for teaching me. To Manu and Roy for all the help with models and real-time. And, of course, to all my colleagues in the lab, for receiving me this well and for making everyday life easier.

To my family, for all the unconditional support and confidence they have always shown to me. To my parents, for listening to me and do anything they could, even in the distance, to Carmen for having the exact advise and to María, for supporting me everyday and taking care of me this well.

Last, but not least, to my friends, for listening to me and worry about me anywhere they are. To the ones in the distance, for all those visits, days back home or call, that could cheer anyone up. And to you all that are here close, for all those "mind-off" moments. Specially to Maria, for all that time spent in EPS, Rodri and MAPS, for always be up to go out and have the right words. And Alvaro, for supporting and putting up with me when it is not that easy.

Agradecimientos

Me gustaría comenzar agradeciendo este trabajo al Grupo de Neurocomputación Biológica (GNB). En primer lugar, a mi tutor, Pablo, por introducirme en el mundo de la Neurocomputación, así como por todas las horas dedicadas y confianza depositada en mi. También a Irene y Rafi, por todas las horas de experimentos y la paciencia enseñándome. Y a Manu y Roy por toda la ayuda con los modelos y el tiempo real. Y, por supuesto, al resto de compañeros por acogerme tan bien en el grupo y por hacer el día a día más fácil.

A mi familia, por tanto apoyo y confianza incondicional que siempre me han dado, y que me siguen dando. A mis padres, por escucharme y hacer todo lo posible aunque fuera a la distancia, a Carmen por tener siempre el consejo adecuado y a María por aguantarme en el día a día y cuidarme tan bien.

Por último, que no menos importante, a mis amigos, por escucharme y estar pendientes de mi desde cualquier sitio. A los que están a la distancia, por esas visitas, vueltas a casa o llamadas, que reaniman a cualquiera, y a los que estáis cerca, por todos esos ratos de despeje tan necesarios. Especialmente a María, por todas las horas en la EPS, a Rodri y MAPS por estar siempre dispuestos a salir y saber decirme siempre las palabras correctas. Y a Álvaro, por apoyarme y aguantarme cuando no era tan fácil.

Contents

List of Figures

1 Introduction

The project proposed for this master thesis addresses the combination of models and electrophysiological experiments in *Lymnaea Stagnalis* for the characterisation of neuronal sequences, as well as the design and implementation of hybrid experiments to address this problem from different perspectives.

This project is divided in two different documents to meet the requirements of the double master degree at EPS-UAM: the one entitled "Experimental and theoretical characterization of CPG activity in *Lymnaea stagnalis*" [Garrido Peña, 2019] and a second one which corresponds to the present document. They both approach the stated problem from different viewpoints. In the first part we discussed the generation and coordination of robust yet flexible sequences, and in this second part we address the recognition of simple sequences in the form of spike trains.

Here on, we will focus on the analysis of neural mechanisms that can produce the recognition of temporal structures in spike input trains by combining subthreshold oscillations and dynamic synapses on a post-synaptic neuron. We will address this problem using both computational models and hybrid circuits that include model synapses and living cells implemented with real-time software technology.

1.1 Hybrid circuits

In the context of Computational Neuroscience, the interaction between classic Neuroscience and Computer Science has contributed important advances for both fields. Nowadays we can appreciate this in many techniques and solutions to health problems involving interactions between a living system and a machine, e.g., prostheses, brain machine interfaces, neural implants, etc [Konrad and Shanks, 2010, Hong and Khan, 2017]. Is in this context where another breakthrough in this multidisciplinary field was developed a few decades ago but keeps being undeveloped to some extent: Hybrid Circuits [Yarom, 1991, Szücs et al., 2000, Pinto et al., 2000, Varona et al., 2001, Le Masson et al., 2002, Amaducci et al., 2019, Reyes-Sanchez et al., 2018]. This technique associates living and artificial neurons and synapses by creating a circuit between these entities in open- or closed-loop protocols. Such interaction results in an effective modification of living dynamics, and it is as it has been set out in several studies, e.g. [Le Masson et al., 2002, Pinto et al., 2000, Norman et al., 2016, Mishchenko et al., 2018, Ambroise et al., 2017, Nishikawa et al., 2019].

Being able to modify neural activity with a hybrid circuit implies many advantages for the research in the nervous system. On the one hand, it is a powerful tool for the understanding of the generation and coordination of neural activity, its associated spatio-temporal patterns and coding and decoding mechanisms. On the other hand, studying the interaction between a living and an artificial system, and developing the techniques that makes it possible, sets a baseline for machine and biological elements interaction, orientated to improve life quality or solve health problems. Therefore, hybrid circuits and close-loop techniques might have many different applications from basic research to biomedical applications like neuroprothesis [Chamorro et al., 2012, Potter et al., 2014, Levi et al., 2018]. In a long-term goal, we can all imagine a future where artificial systems replace damaged brain areas or enhance the performance of the nervous system.

For a better understanding on what hybrid circuit actually are, we will discuss a bit more in detail the elements for their implementation in an *in vitro* preparation. On the one hand, there is a living part, which is usually a neural system from which activity can be recorded and where stimulation can be delivered. There are several techniques for acquiring the living system signals as well as different biological entities from where they can be acquired. They could be cultured *in vitro* as a neuronal network and be electrically accessed through Micro-Electrode Arrays (MEAs) [Frega et al., 2012]; or they could also be neural systems isolated from the biological entity, and connected using some intra- or extra-cellular techniques.

On the other hand, the artificial element in the circuit is usually a neural model running on some hardware

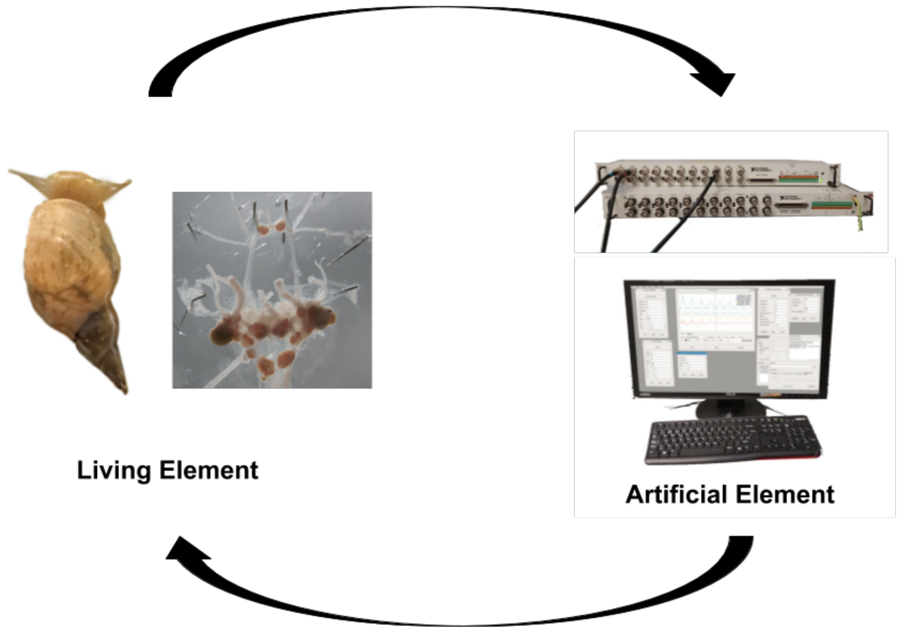


Figure 1: Hybrid circuit basic schema: neural activity is recorded from the living element, then used in a model to compute a response typically involving model neurons and/or synapses and the resulting output is delivered back into the living element. This can be implemented in open or closed-loop versions and involves fast cycles typically implemented in real-time software platforms [Amaducci et al., 2019, Reyes-Sanchez et al., 2018].

.

or software based system, e.g. FPGAs or a real-time capable computer, respectively. Neural models are mathematical descriptions of the neural activity [Torres and Varona, 2012]. They simulate neural activity, typically described as a membrane voltage which results from the combination of different ionic channels which represent intrinsic properties of neuron conductances. Since the first models were developed [Hodgkin and Huxley, 1952, Hindmarsh and Rose, 1984], there have been many different approaches to represent neural activity at different levels of abstraction and accuracy regarding the description of specific neuron types, e.g. see [Komendantov and Kononenko, 1996, Vavoulis et al., 2007, Izhikevich, 2003].

Figure 1 displays a schematic representation of a typical hybrid circuit design. Neural activity is recorded from the living element, then used in a model to compute a response which typically involves model neurons and/or synapses and the resulting output is delivered back into the living element, see also [Reyes-Sanchez et al., 2018]. Closed-loop approaches typically rely on real-time event detection for the characterization of key events in neural dynamics which also define when the stimulus needs to be delivered to the living cells [Arroyo et al., 2013, Varona et al., 2016].

Hence, neuron and synapse models can be combined with living neurons either as an additional part of the circuit or as a source for controlled inputs such as a train of pre-synaptic spikes delivered with a model synapse. Depending on how is this interaction is built and the role of the artificial neuron(s) and/or synapses on the circuit, there are two types of hybrid-circuit experiments: closed-loop and open-loop. In a closed-loop protocol, the model neuron becomes part of the circuit through a bidirectional connection. Thus both the artificial and the living neuron receive information from their counterpart and change their behaviour as a result of this interaction. In other words, both entities, living and artificial, receive a synaptic input coming from the other cell. As opposed to this approach, the open-loop technique, implies a monodirectional input typically from a model neuron and/or synapse to the living cell.

In this project, we are going to focus on software based solutions for hybrid circuits. The living part corresponds to a neuron from *Lymnaea stagnalis* isolated neural system, whose morphology and circuitry is summarised in the complementary document to this project [Garrido Peña, 2019]. Therefore, the electrophysiology techniques used to make the connection possible in hybrid experiments will be implemented by intracellular recordings with sharp electrodes. Hybrid circuits in this modality have their origins in the Dynamic Clamp concept of stimulation [Robinson and Kawai, 1993, Sharp et al., 1993]. When perturbing a neuron in an experimental setup, there are different possible approaches, depending on the elecrophysiological technique used to deliver input. Traditionally, there were two main techniques: voltage- and current-clamp, fixing values of voltage and current, respectively (more info about this is available in part 1 of this project [Garrido Peña, 2019]). However, in 1993 Dynamic Clamp was presented as a novel technique. The novelty of this technique resided in the possibility of applying a dynamic and not fixed current that might modify neural dynamics in a less restricted way. Since the current is set by a computer connected to the *in vivo* or *in vitro* sample, usually through a DAQ board, it is possible to implement artificial ionic or synaptic contuctances [Nowotny and Varona, 2015]. This concept can also be generalised with the use of neurotransmiter or neuromodulator microinjection, or mechanical stimulation [Muniz et al., 2008, Chamorro et al., 2009, Chamorro et al., 2012].

1.2 Real-time software technologies

When connecting a living neuron to an artificial one, the latter is supposed to adapt to the living activity, e.g. firing with a realistic timing and phase. Neurons' activity produces spikes in the order of milliseconds, which means the software used to run the model simulation, must be solved at the DAQ rate of 10 KHz or more, i.e., the software solution must run in hard real-time for a realistic interaction or stimulation.

A real-time system is usually conceived in the context of some real element setting a time reference, e.g. a human pressing some key, a train detecting a red light or (in the case of this project) a neuron generating action potentials. Therefore, a machine will behave in real-time as long as it can adapt precisely to the real element time basis.

Since these systems are associated with something real, they mainly depend on the sampling rate of the acquisition system, which is the period within it is conversing some analogical input into some digital output to be processed. This is why, choosing the sample rate is a key element, and thus it is not only important for a duly fine digitalisation in time and shape.

It is because of this conversion and the necessary computing processing that, when talking about real-time, there will always be some delay involved. The key is to adjust the system to the requirements of the temporal precision so that this delay does not compromise it. For instance, when we use a keyboard (a human computer interaction) as we press a key, the corresponding letter shows up in the screen, for us it seems real time, but the two actions have not occurred simultaneously. The time the computer takes to process the command implies a delay, but, since it has happened in less than a second, it is negligible for the human. However, this less than a second delay, might be a problem in other real-time systems, not only for the perception of the delay, but also because it can imply missing data, for example in heart rating or seism detection systems, where not having an adequate temporal precision results in catastrophic results.

Is this context, we can classify real-time systems in hard and soft real-time systems. In soft real-time, information loss or a significant delay does not have really important consequences. On the other hand, in hard real-time systems, a delay in the response or information loss caused by it might have very negative consequences and, thus, is prevented. In this project, on hard read-time software implementations, not only because recording and stimulation sampling must kept a strict low delay to avoid losing signal information, but also because all computations need to be performed at the living neuron's activity time.

1.3 System real-time adaptation

Hence, for hybrid circuits performance, it is necessary to have an operating system (OS) able to adapt precisely to real-time necessities, i.e., to neuron activity time [Amaducci et al., 2019]. Adapting an OS to real-time usually is based on scheduling algorithms (clock-driven, priority-driven, etc.). The main goal is to schedule all the processes in order to perform each of them at a precise time-interval, specifically the ones requiring the real-time.

In a general-purpose operating system (OS), as well as in a real-time OS different functions may have distinct temporal needs. An OS usually has a processing based on some key elements: threads, system calls (interruptions), clock and scheduler [Liu, 2000]. Every time the main system thread receives an interruption from some task requiring processor capacity, the scheduler "organises" the queue regarding its priority and the corresponding scheduling algorithm. A thread is created, stopped or restarted, depending on the specific necessity.

Since a real-time system is not about computing fast, but computing with temporal precision, a real-time OS must handle these priorities, so that the real-time tasks are executed at the exact moment, and thus are not interrupted by any other irrelevant (in real-time terms) task. Hence, in the context of the hybrid circuits developed in this project, the OS must adapt to the acquisition board which operates at 10kHz, being essential for the real-time to perform all necessary actions for the interaction with the living cell in that period of time.

However, for hard-real time systems like this, it is not enough with this priority modifications. The operating system itself must be reconfigured, since, even in the most flexible OS (at a super user level) the process of modifying priorities and processing scheduling does not allow a complete control, but typically just the adjusting of its performance.

In the context of precise temporal computing and scheduling timing, operating systems focused on real-time performance are known as Real-Time Operating Systems (RTOS). This kind of systems have an strict scheduling algorithm, since, in contrast to general purpose OS, they are not aimed to run a wide variety of applications but only a few of them. So there is no need for adapting the scheduling and priorities to the wide range of applications that the user might need, but just a few of them, although very precisely. Two key concepts when talking about RTOS are interrupt latency and jitter [Liu, 2000]. Interrupt latency is the time delay between an interruption occurs and the response of the processor. Only when the clock frequency is known, it is measured in $\mu seconds$, otherwise we can only talk about number of cycles. Therefore, this delay usually depends on what tasks the processor is handeling at the same time, and the scheduling algorithm used.

On the other hand, jitter is defined as the difference between the maximum and minimum latencies. Hence, it represents the variability of the latency, being a general measure of how this latency affects the system.

The hybrid tools that we will discuss throuhout this document use this kind of real-time operating systems to coordinate living activity recording and the computing of artificial models in hybrid circuit configurations.

1.4 The need of neuron adaptation in hybrid circuits

During a hybrid experiment there are many factors involved. Even though real-time is an essential part, it is not the only factor that needs to be taken into account for a good implementation of the experiment. When combining living and artificial neurons, there must be an adaptation between them to achieve a realistic interaction [Reyes-Sanchez et al., 2018].

The easiest solution for adapting them is modifying model parameters until they fit the living neuron, since the model is the most flexible one in terms of amplitude, and amplitude scale modification. This is an essential task prior to the experiment, since it conditions how well the model will adapt to the living signal, and consequently, affects the experiment's correct course. It is possible to do this by hand-tuning the parameters, however this is a tedious approach for the experimentalist, since a first initial adjusting is not usually enough. The living neuron may vary its activity during the experiment in terms of amplitude, offset, drift, etc. and, moreover, different models might be used during the experiments.

Therefore, some adaptation algorithms have been developed to make hybrid experiments easier and more precise [Reyes-Sanchez et al., 2018]. These set of applications are included in a software platform [Amaducci et al., 2019], performing successful automatic adaptations during the experiments.

Based on those algorithms we can identify which are the most relevant scaling elements to take into account when designing and carrying out hybrid experiments.

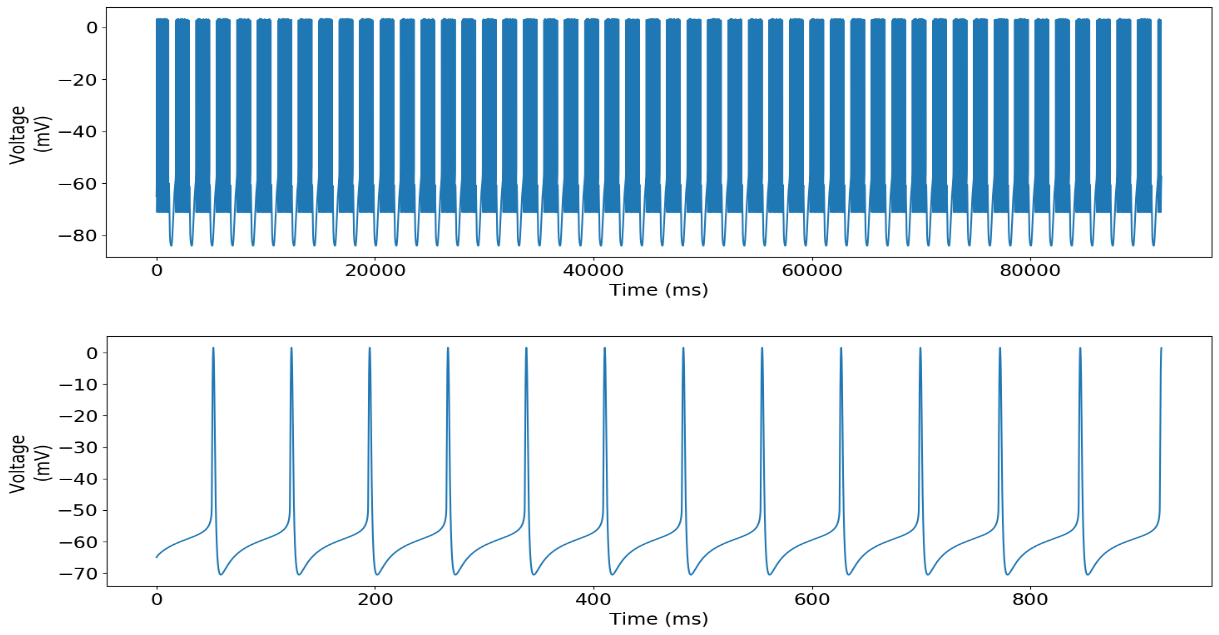


Figure 2: Neuron simulation traces with same number of iterations but different integration time steps.

On the one hand, we find time scaling. Equations conforming computational models typically simulate activity depending on an integration time step (dt). Therefore, with a given number of iterations but different time step, the model will simulate different time intervals. This is illustrated in figure 2 with different time steps (dt= 0.1 and dt=0.001) and the same number of iterations for integrating the neuron model used in the complementary document of this project [Garrido Peña, 2019]. Apart from that, not all models have the same iterations-time relation, so for a same dt in two different models, the simulation might be completely different. Therefore, to adapt it to a living neuron the DAQ board sampling rate and this parameter must be taken into account. In Reyes et al. the algorithm defined to adapt this parameter is based on the events analysed, specifically bursts. Hence, the model neuron's burst must have the same duration than the living neuron bursts.

On the other hand, amplitude and offset are key issues, since not all models generate the same amplitude nor all living neuron recordings. The offset can also vary in each preparation. The algorithm proposed by Reyes et al. for this adaptation calculates the range of the amplitude of a neuronal event in the living cell, so that the resulting voltage of the artificial neuron can be adapted. Following the same logic, current to the living neuron can be also adjusted. Even though there are more elements to take into account during a hybrid experiment, these are the essential ones for the implementation of hybrid experiments.

1.5 Aim of this study

This study addresses the design of a hybrid experiment to test the interaction of dynamic synapses and intrinsic subthreshold neuronal oscillations in shaping the response to spike trains with different temporal structure. For this purpose, a conductance-based model with subthreshold oscillations will be characterised in the context of this type of stimulation to map input/output non-trivial relationships. The experiment will be then implemented in a real-time software tool, RTXI, to induce subthreshold oscillations into a quiescent living neuron and to perform the analysis in living neurons.

2 Methods

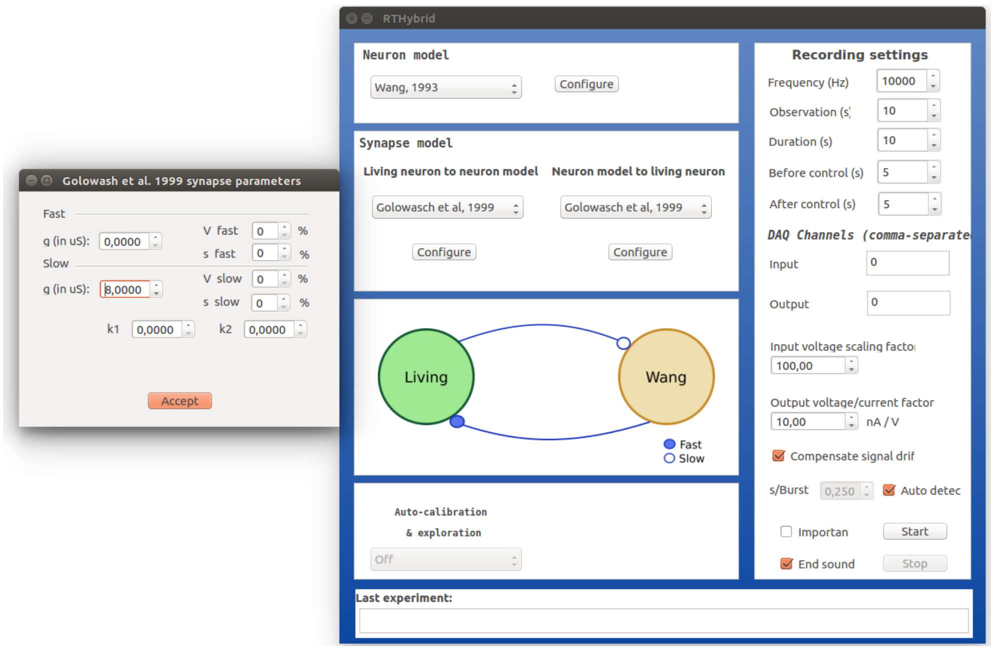
2.1 Tools to implement hybrid circuits

Once we have discussed the basics for hybrid circuit construction regarding the necessity of real-time, set-out, and the adaptation of the living and artificial neuron parameters, let us focus on the different tools to implement this paradigm.

On the one hand, there are several platforms for building hybrid circuits (for a review see [Amaducci et al., 2019]). As a general classification, we could differentiate between hardware and software solutions. Hardware solutions, are fixed circuits that simulate neural behaviour and their input/output is connected to the living entities. These, as any other hardware systems, are less flexible than software solutions, for instance, in terms of on-the-fly parameter tuning. Even though modern hardware solutions such as Arduino platforms reduce this problem, they have nothing to do compared with the flexibility of a software-based solution.

Recently, different software solutions have been developed, but here we are going to focus on two software platforms based in RTOS: RTHybrid [Amaducci et al., 2019] and RTXI [Patel et al., 2017]. They both run in real-time operating systems and they both are validated solutions for hybrid experiment implementations. To obtain data from the living neuron, a Data Acquisition Board (DAQ) works as an intermediate stage transforming the living neuron's signal from the amplifier to the computer and back. The DAQ is handled by these software platforms.

2.1.1 RTHybrid


RTHybrid

Neuron model: Wang, 1993

Synapse model
Living neuron to neuron model: Golowasch et al, 1999
Neuron model to living neuron: Golowasch et al, 1999

Auto-calibration & exploration: Off

Recording settings
Frequency (Hz): 10000
Observation (s): 10
Duration (s): 10
Before control (s): 5
After control (s): 5
DAQ Channels (comma-separate
Input: 0
Output: 0
Input voltage scaling facto: 100,00
Output voltage/current factor: 10,00 nA / V
[✓] Compensate signal drif
s/Burst: 0,250 [✓] Auto detec
[ ] Importan
[✓] End sound

Last experiment:

Golowash et al. 1999 synapse parameters
Fast
g (in uS): 0,0000
V fast: 0 %
s fast: 0 %
Slow
g (in uS): 8,0000
V slow: 0 %
s slow: 0 %
k1: 0,0000
k2: 0,0000

Figure 3: RTHybrid interface screenshot

RTHybrid [Amaducci et al., 2019] is a program prepared for running on real-time systems, particularly on Preempt-RT and Xenomai 3. This tool is focused on an easy adaptation for the living and artificial neurons, both in mono- and bi-directional connections. This is possible due to its library of models which contains different neural paradigms. Hybrid circuits are easily implemented by selecting each model with its corresponding hand-tuned parameters, if necessary. The system allows as an option to load preset parameters from an xml file. In addition, one of the greatest advantages of RTHybrid is the automatic adaptation of the model entities to the living system. After a few control recording, living neuron characteristics are analysed and the model is tuned in terms of time and amplitude scales. The system can also take into account the neuron drift dynamically. An example of its interface is displayed in figure 3.

Since this program is not thought to be a tool for neural activity visualisation during the experiment, it does not provide real-time plotting of the signals being recorded. However, it saves the results of both model and living recording into files, which can be easily plotted offline.

Therefore, when the aim is to connect model neurons to a living neural system, RTHybrid is a nice solution because of its different model alternatives, the associated flexibility to build the hybrid interaction, as well as the automatic adaptation of both living and artificial cells. All these tools considerably reduce the set-up time of the experiment.

2.1.2 RTXI

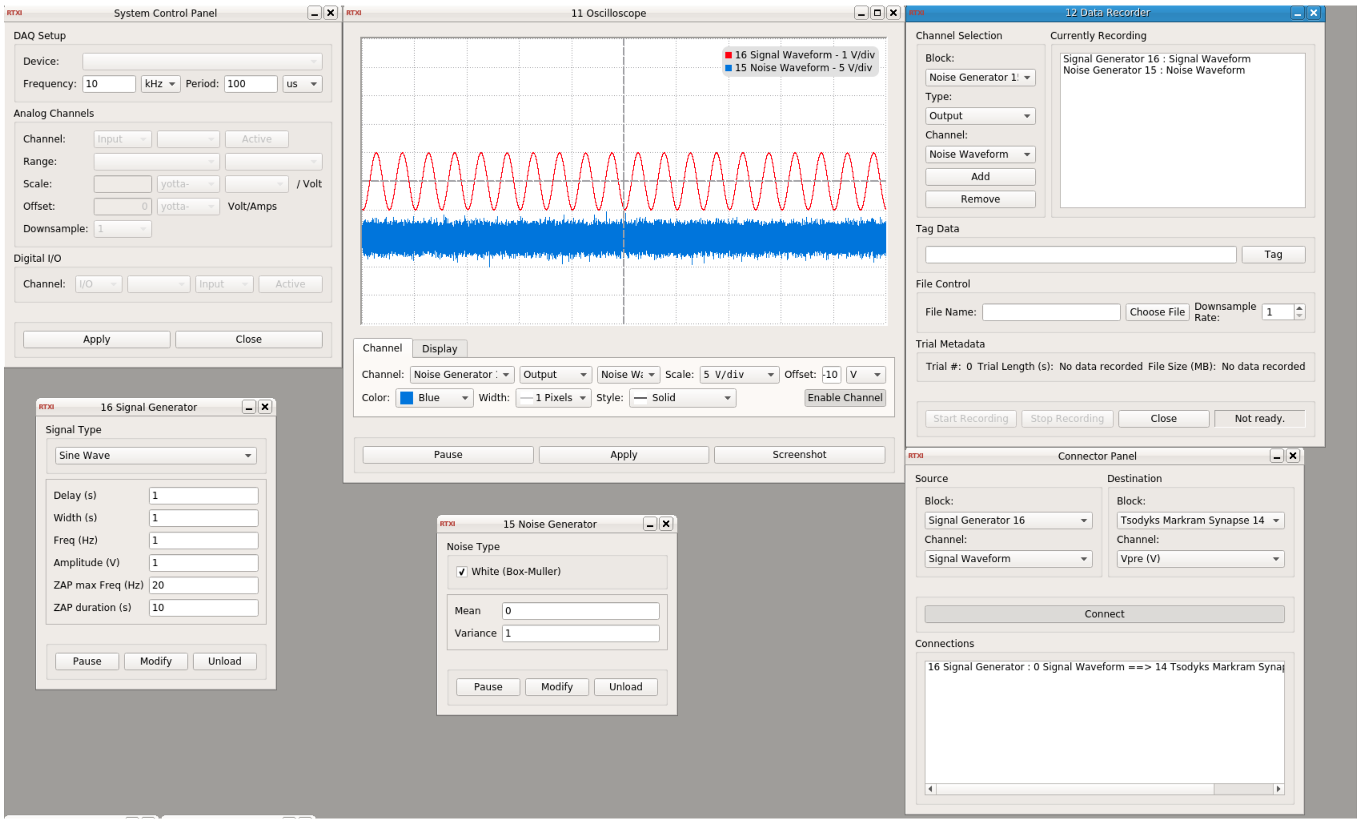
RTXI [Patel et al., 2017] is another tool oriented to the construction of hybrid experiments based on real-time solutions. It also runs on the real-time framework Xenomai. The strength of this tool is its module-based interface. It has some default modules which allow a basic functionality of the system: oscilloscope, connector, data recorder, etc. In addition, using a set template (open source available at `https://github.com/RTXI/plugin-template`), modules can be implemented and included in the system for specific experimental needs. This requires some knowledge of C++ and interaction with the interface, since the template is only a basis for the later self-implementation.

The data acquired from the living neuron by a DAQ board can be shown on-the-fly on the oscilloscope panel, along with some other outputs from self-implemented modules which can be set to interact with the neuron. For this interaction, modules must have been previously connected by the connector panel, which pipes a module output to some other module input. An example of some of these modules is shown in the screenshot displayed in figure 4.

Hence, this real-time software solution is a good option when designing whole new experiments, since it can easily be adapted to neuron, synapses or any other activity setting.

Using the Recording panel it is possible to save experimental data. For this purpose RTXI saves a binary file using the HDF5 format. The aim of using this format is to favor data accessing, since it is organised hierarchically by chunks that can be accessed with no need of loading the whole file. This data can be handled by python using a library. The data is classified in Trials with all the associated information included.

For this project, RTXI will be used in the experiments, due to its adaptability by modules and since some of the modules oriented to the development of this experiment are already available in github (`https://github.com/GNB-UAM/triple_pulse_stimulator` ; `https://github.com/GNB-UAM/tsodyks_markram_synapse`).


System Control Panel

DAQ Setup

Device:

Frequency: 10 kHz Period: 100 us

Analog Channels

Channel: Input Active

Range:

Scale: yotta- / Volt

Offset: 0 yotta- Volt/Amps

Downsample: 1

Digital I/O

Channel: I/O Input Active

Apply Close

11 Oscilloscope

16 Signal Waveform - 1 V/div
15 Noise Waveform - 5 V/div

Channel Display

Channel: Noise Generator 1 Output Noise Wa Scale: 5 V/div Offset: -10 V

Color: Blue Width: 1 Pixels Style: Solid Enable Channel

Pause Apply Screenshot

12 Data Recorder

Channel Selection

Block: Noise Generator 1

Type: Output

Channel: Noise Waveform

Add Remove

Currently Recording

Signal Generator 16 : Signal Waveform
Noise Generator 15 : Noise Waveform

Tag Data

Tag

File Control

File Name: Choose File Downsample Rate: 1

Trial Metadata

Trial #: 0 Trial Length (s): No data recorded File Size (MB): No data recorded

Start Recording Stop Recording Close Not ready.

16 Signal Generator

Signal Type

Sine Wave

Delay (s) 1

Width (s) 1

Freq (Hz) 1

Amplitude (V) 1

ZAP max Freq (Hz) 20

ZAP duration (s) 10

Pause Modify Unload

15 Noise Generator

Noise Type

✓ White (Box-Muller)

Mean 0

Variance 1

Pause Modify Unload

Connector Panel

Source

Block: Signal Generator 16

Channel: Signal Waveform

Destination

Block: Tsodyks Markram Synapse 14

Channel: Vpre (V)

Connect

Connections

16 Signal Generator : 0 Signal Waveform ==> 14 Tsodyks Markram Synap

Figure 4: Screenshot of RTXI interface with example modules.

2.2 Hybrid circuit to explore neuronal input-output relationships

Temporal discrimination of input trains is a key information processing function carried out in many different neural systems [Latorre et al., 2006, Torres et al., 2019]. Intrinsic subthreshold oscillations can implement information discrimination mechanisms in single cells [Llinas, 1988] and powerful coordination mechanisms in large networks [Varona et al., 2002, Latorre et al., 2013]. On the other hand, dynamic synapses allow channel-specific synaptic modulation [Tsodyks and Markram, 1997]. Modeling studies have hypothesised that intrinsic neuronal properties, in particular subthreshold oscillations lead to the emergence of nontrivial single-cell input-output preferences, e.g., preference towards decelerating vs. accelerating spike trains [Baroni and Varona, 2007, Baroni et al., 2010]. It has also been shown that synaptic plasticity [Baroni and Varona, 2010] and that short-term synaptic dynamics [Latorre et al., 2016, Torres et al., 2019] can be combined with the dynamics of subthreshold oscillations to further tune this nontrivial preferences in a channel specific manner.

The aim of this hybrid experiment is to setup an experimental protocol to test the effect of intrinsic subthreshold oscillations and its interaction with dynamic synapses for the study of non-trivial responses to pre-synaptic inputs trains and thus assess the model theoretical predictions [Torres et al., 2019].

2.2.1 Hybrid set up. *Lymnaea Stagnalis* as a system for hybrid circuit implementation

As we have seen so far, there are several things involved in a hybrid experimental set-up. An essential element is the living system where neural activity will be recorded. The living element that has been chosen for this work is *Lymnaea Stagnalis*, since the neural system isolation is easier than in other systems and its neurons are rather big. A detailed analysis of this mollusc can be found in the complementary document of this project [Garrido Peña, 2019], as well as the description of different electrophysiological techniques to record neural activity.

Hence, in this experiment a mono-directional artificial connection will be implemented on neurons of *Lymnaea Stagnalis*. For this purpose, intracelullar techniques will be used, introducing two microelectrodes into the same neuron. One of them will record neural activity and the other one will inject the stimulation in the form of an artificial synaptic current from the model modules.

The neuron selected for the experiment would ideally be a fairly silent one, except from the desirable presence of subthreshold intrinsic activity. For each experiment, the seek of this neuron will be performed in the parietal ganglion, as the neurons here located are usually not very active and subthreshold oscillations have been sighted. Figure 5 shows an example of the location of this ganglion within the complete *Lymnaea* neural system, as well as the neurons present in this ganglion in a image obtained with a higher magnification in the microscope.

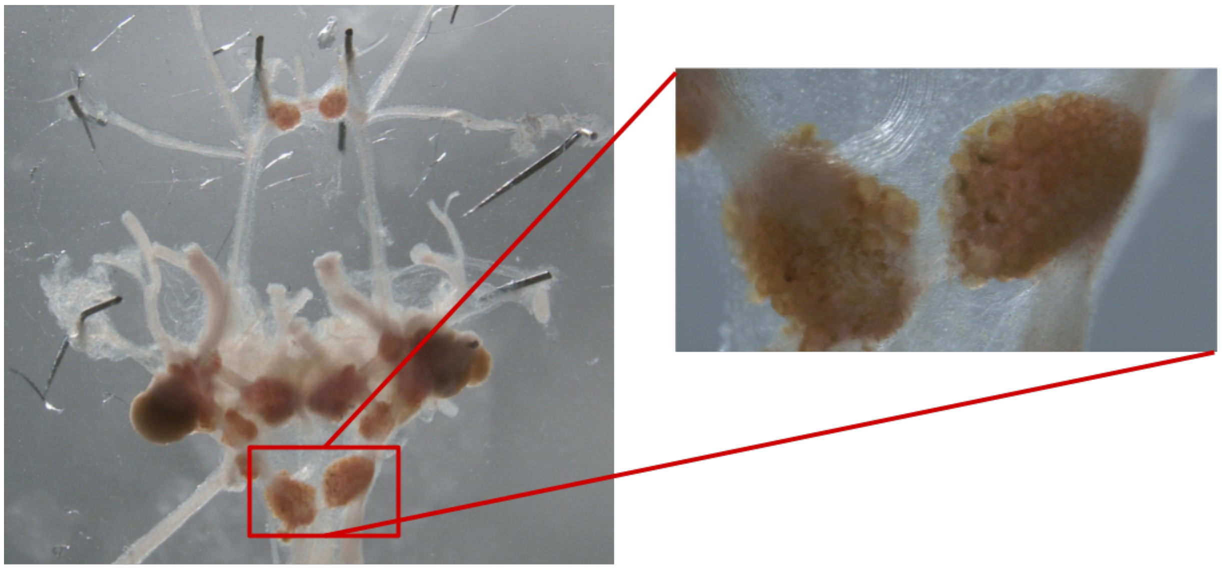


Figure 5: *Lymnaea* parietal and visceral ganglia.

The neural activity recorded by a sharp electrodes in the neuron is processed by the amplifier which sends the amplified signal into a DAQ board connected to a computer. There, the signal is processed by RTXI which reads the analog input received by this DAQ. The experimental set-up information show is illustrated in figure 6.

In RTXI, distinct modules are used to process this signal and perform the required computational operations necessaries to send the synaptic stimulation and evoke a response from the neuron. For this purpose, RTXI sends the input signal back to the DAQ, which is connected to the amplifier current input. There, the input is parsed in the proportion of 10A/V and injected through the electrodes into the neuron.

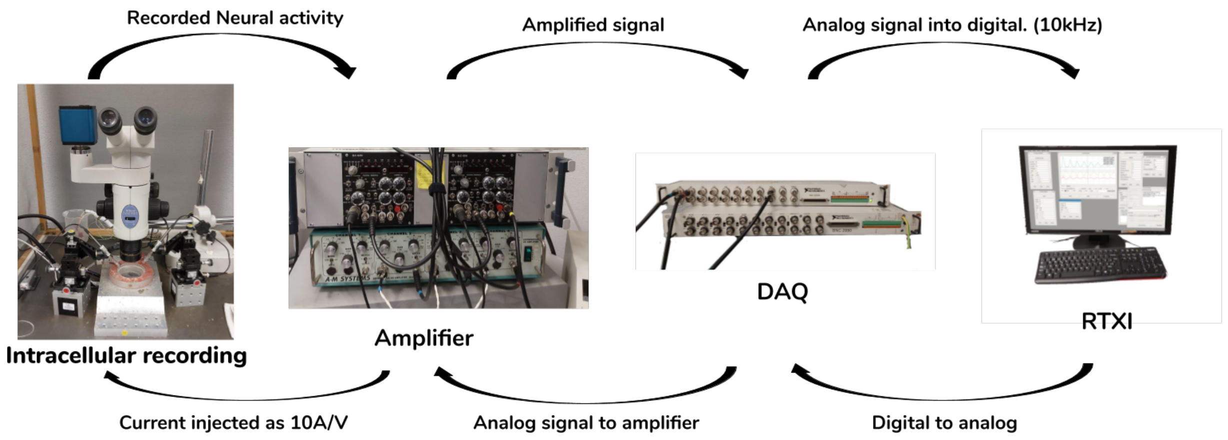


Figure 6: Hybrid circuit set-up diagram.

2.2.2 Experimental design

Having a candidate neuron for the experiment, the input signal to be applied in the form of a synaptic current correspond to triplets of presynaptic pulses of two types: accelerating and decelerating. This two types of stimuli are illustrated in figure 7. Their definition is conceptually simple, a triplet is accelerating when $ISI_1 > ISI_2$, being the ISI the inter spike interval, i.e. the time interval between pulses; when the opposite occurs, $ISI_2 > ISI_1$ the triplet is decelerating. Input/output relations are called non-trivial in this document when there is a response to a decelerating input or an absence of response to an accelerating one. Nontrivial responses are related to the concept of complex information discrimination [Baroni and Varona, 2007, Torres et al., 2019]. Trivial input/output relations correspond to a spike response to accelerating triples and absence of response to decelerating ones, which is the typical behaviour of integrator neurons with pure temporal discrimination capabilities.

This simple triplet pulse could be applied using a simple classic Dynamic-Clamp. However, this experiment is not as simple as injecting three pulses of current into the cell. Besides the need to develop a new module or use an existing module for the generation of 3 pulses, the effect of this pulses needs to be converted into the current corresponding to the effect of the dynamic synapse. Thus, the pulses will be processed and converted into a synaptic current by the model described by [Tsodyks and Markram, 1997], which represents the temporal evolution of the post-synaptic effect on the neuron that receives the input.

Moreover, the candidate neuron for this experiment may not have the ideal conditions for this experiment, i.e., intrinsic subthreshold oscillations or be silent in the absence of stimuli. Hence, it is an option to use the possibilities of hybrid technologies to induce such behaviour by means of additional artificial currents which would add this missing characteristics to the neuron.

In this section we are going to go into detail through all different parts of the set-up of this experiment. We will also describe the models needed to design and implement the hybrid circuit.

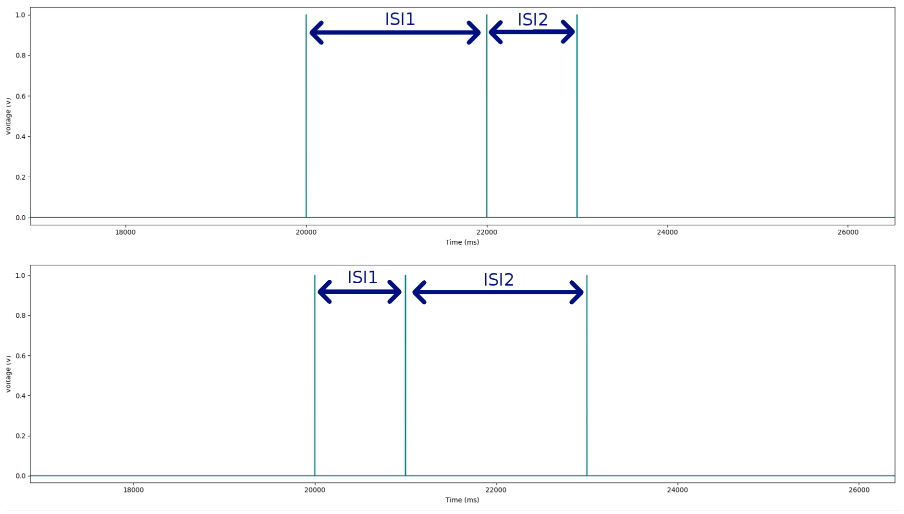


Figure 7: Accelerating (top) and decelerating (bottom) triplets

2.2.3 The search for non-trivial responses

As we have just mentioned, the effect of these triplets might be altered in the presence of intrinsic subthreshold oscillations [Torres et al., 2019], leading to non-trivial responses obtaining a response in form of an action potential after a decelerating triplet, and silence after an accelerating one.

To obtain this response, a right configuration of ISI_1 and ISI_2 must be set. As it is explained in Torres et. al., non-trivial responses may occur when the shortest ISI part of the triplet (ISI_1 in case of a decelerating and ISI_2 in case of an accelerating one) are approximately half of the subthreshold oscillations period.

Therefore, to obtain non-trivial responses in the living system, it is important to go through many different possible ISIs configurations to test this effect not only because of the evolution of the subthreshold oscillations, but also because of the timing of the occurrence of the pulses.

2.2.4 Hybrid experiment modules for RTXI

As we have already mentioned, RTXI allows for an easy adaptation of customised modules. This is the case of the two modules applied here for the execution of the experiment.

Triple Pulse Stimulator

On the one hand, for the generation of triplets of pre-synaptic pulses, there is a free software module available (`https://github.com/GNB-UAM/triple_pulse_stimulator`).

In figure 8 there is a screenshot of the module, where we can see all the tunable parameters.

The functionality of this module is conceptually easy. It has two time references, corresponding to each ISI, which are increased after each *recovery time* selected by the experimentalist from a specified *step* value. Therefore, each ISI increases gradually its value from the *T1 initial* and *T2 initial*, until reaching *T1 max* and *T2 max*, respectively.

This set of parameters allow for a flexible solution to go through different ISIs combinations during the experiments.

Tsodyks Markram Synapse

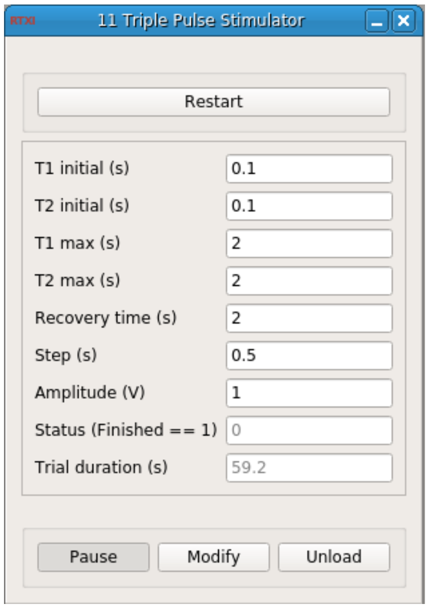
As another key element of the experimental set-up is the implementation of the Tsodyks-Markram dynamical synapse [Tsodyks and Markram, 1997], as we have already mentioned. A module implements this synapse using the following equations, as defined in [Torres et al., 2019]:

$$
\begin{aligned}
\frac{dx}{dt} &= \frac{1-x-y}{\tau_{\text{rec}}} - U(t_n)x(t_n)\delta(t-t_n) \\
\frac{dy}{dt} &= \frac{-y}{\tau_{\text{in}}} + U(t_n)x(t_n)\delta(t-t_n) \\
\frac{dU}{dt} &= \frac{(\mathcal{U}-U)}{\tau_{\text{fac}}} + \mathcal{U}[1-U(t_n)]\delta(t-t_n\ )
\end{aligned}
\qquad (1)
$$

where $x(t)$ represents the fraction of recovered neurotransmitters, $y(t)$ the fraction of active neurotransmitters, $U(t)$ the fraction of released neurotransmitters after the arrival of an action potential, and t_n is the timing of the most recent synaptic event. In this description, as explained in [Torres et al., 2019] et al, $x(t_n)$ and $U(t_n)$ are the values of the corresponding variables immediately before the arrival of the most recent pulse at t_n, and $\mathcal{U}$ represents the modelled fraction of released neurotransmitters at rest.

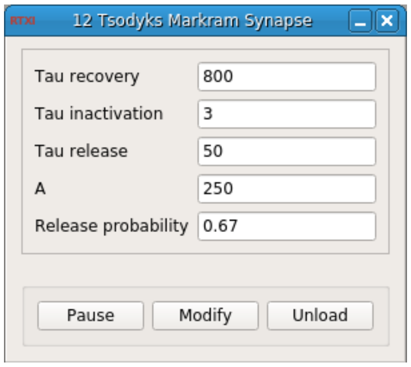
The parameters shown in the corresponding GUI are the taus from each (x,y,U), along with the release probability ($\mathcal{U}$) and amplitude (A in GUI) which corresponds to the maximal synaptic strength, since the final current is defined as $I_{\text{syn}}(t) = \mathcal{A}_{\text{syn}}y(t)$.

Figure 9 shows a screenshot of this module.



T1 initial (s)	0.1
T2 initial (s)	0.1
T1 max (s)	2
T2 max (s)	2
Recovery time (s)	2
Step (s)	0.5
Amplitude (V)	1
Status (Finished == 1)	0
Trial duration (s)	59.2

Figure 8: Triple pulse stimulator module.



Tau recovery	800
Tau inactivation	3
Tau release	50
A	250
Release probability	0.67

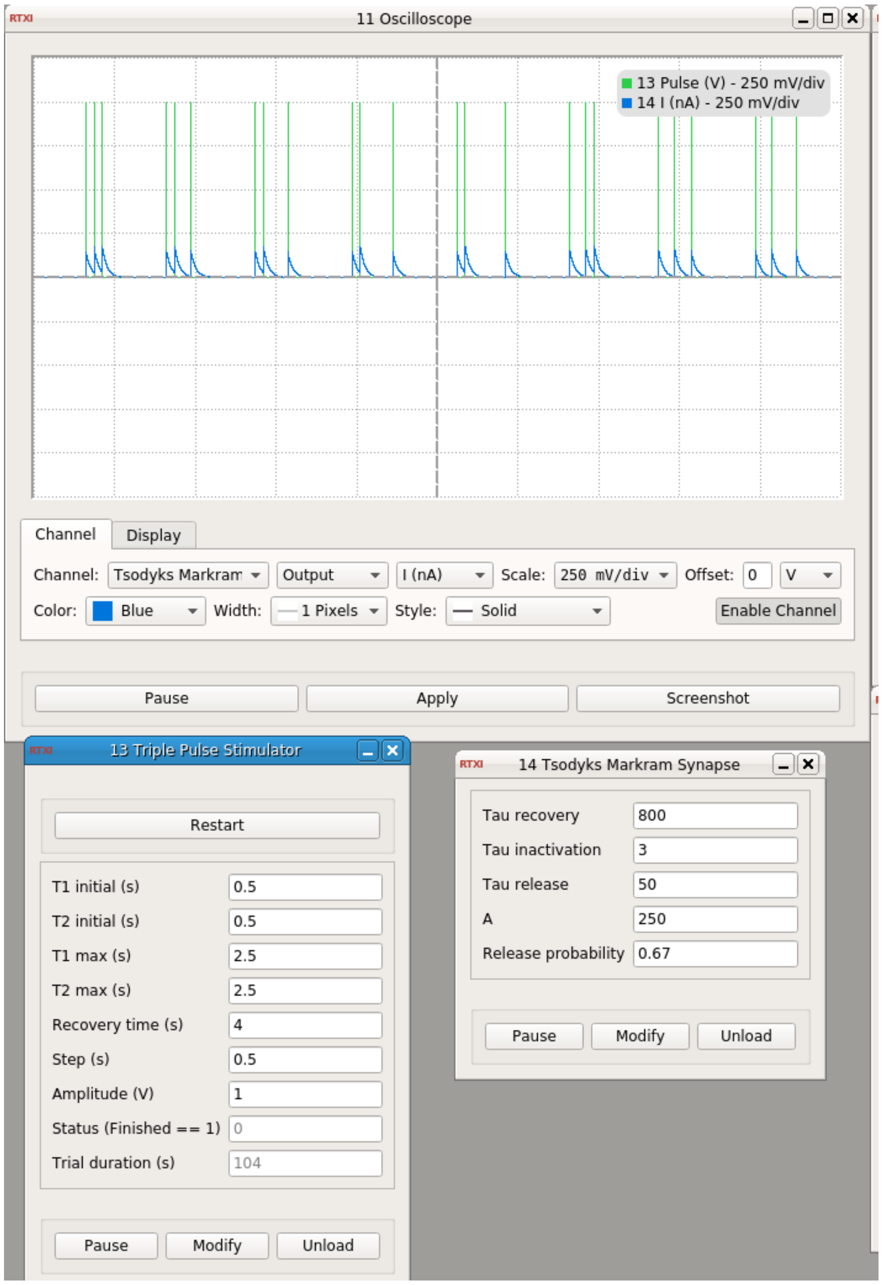
Figure 9: Tsodyks Markram Synapse.

In this way, from the received synaptic input, a synaptic current will be generated. The way of joining these modules with the living neuron is using the Connector System module available in RTXI. Here we can select the parameters from the modules declared as INPUT or OUTPUT and connect each other generating internally a pipe between an OUTPUT and INPUT. In this case:

1. Triple pulse Output → Tsodyks-Markram Synapse Input

2. Tsodyks Synapse Output → Analog output (living signal from DAQ)

An example of the activity jointly generated by these two modules is shown in figure 10.



11 Oscilloscope

🟩 13 Pulse (V) - 250 mV/div
🟦 14 I (nA) - 250 mV/div

Channel: Tsodyks Markram | Output | I (nA) | Scale: 250 mV/div | Offset: 0 | V

Color: Blue | Width: 1 Pixels | Style: Solid | Enable Channel

13 Triple Pulse Stimulator

T1 initial (s)	0.5
T2 initial (s)	0.5
T1 max (s)	2.5
T2 max (s)	2.5
Recovery time (s)	4
Step (s)	0.5
Amplitude (V)	1
Status (Finished == 1)	0
Trial duration (s)	104

14 Tsodyks Markram Synapse

Tau recovery	800
Tau inactivation	3
Tau release	50
A	250
Release probability	0.67

Figure 10: Pulses and the synaptic effect produced in RTXI interface. Pulses in green and synapse in blue.

2.2.5 Model approach

As we have already seen along this document, and its complementary one [Garrido Peña, 2019], models simulating neuron activity play a crucial role in computational neuroscience. Since these models reproduce reliably neural behaviour, simulating first the experiment performed over the living neuron in an artificial neuron brings in a reference on what is to be expected. Therefore, in order to have some reference of the effects of different pulses and synapse configurations, we will first perform the experiment using a model with subthreshold oscillations [X.-J. Wang, 1993].

In this model, the combination of a persistent sodium current and a slowly inactivating potassium current turns into a subthreshold activity when parameters are properly tuned. These equations are adapted to the neural activity generated by classical I_{Na} and I_K currents present in the Hodgkin Huxley [Hodgkin and Huxley, 1952] formalism, along with I_L and the injected current I_{app}. These currents generate the action potential as described in the following equation.

$$C_m \frac{dV}{dt} = -g_L(V - V_L) - I_{NaP} - I_{KS} - I_{Na} - I_K + I_{app} \tag{2}$$

Classic equations from Hodgkin Huxley are defined by:

$$I_{Na} = g_{Na} m_\infty^3 * (V - V_{Na}) \tag{3}$$

$$I_K = g_K n^4 (V - V_K) \tag{4}$$

The equations for I_{NaP} and I_{KS} (currents generating subthreshold oscillations) are defined as follows:

$$I_{NaP} = g_{NaP} m_\infty(V) * (V - V_{Na}) \tag{5}$$

$$I_{KS} = g_{KS} m(ph_1 + (1-p)h_2)(V - V_{KS}) \tag{6}$$

In NaP, the gating variable maintains its steady function, whereas the KS variables are dynamical following an updating gating variable function which implies some τ activation value. In the particular case of m in I_{KS}, τ_m value has a direct effect on rhythmic frequency, increasing as it increases its value. Besides the results reported in Wang et al. [X.-J. Wang, 1993], figure 11 illustrates an example for two different values of τ_m, 30 and 15 (top and bottom panels, respectively).

The oscillation is generated due to the opposition of the 2 channels, as shown in figure 12. Current I_{NaP} stimulates activation of the action potential with a 8 range current, whereas I_{KS} current, depolarises the neuron at the same time with a higher amplitude (range of 25). This way a wave is generated even when there is no spiking nor synaptic input.

Therefore, having this model as an example of subthreshold oscillations, a theoretical basis can be set for the model in two directions. On the one hand, there can be performed simulations of the effect of the pulses in combination to Tsodyks-Markram synapse. Having a reference to compare the experimental results with.

On the other hand, the channels generating the subthreshold oscillations are good candidates to generate it on silent neurons found in *Lymnaea* too. For this purpose, these channels will be adapted to the neuron as RTXI modules.

The model will be implemented in C++ for a fast computation of the simulation and for a easier subsequent adaptation to RTXI modules.

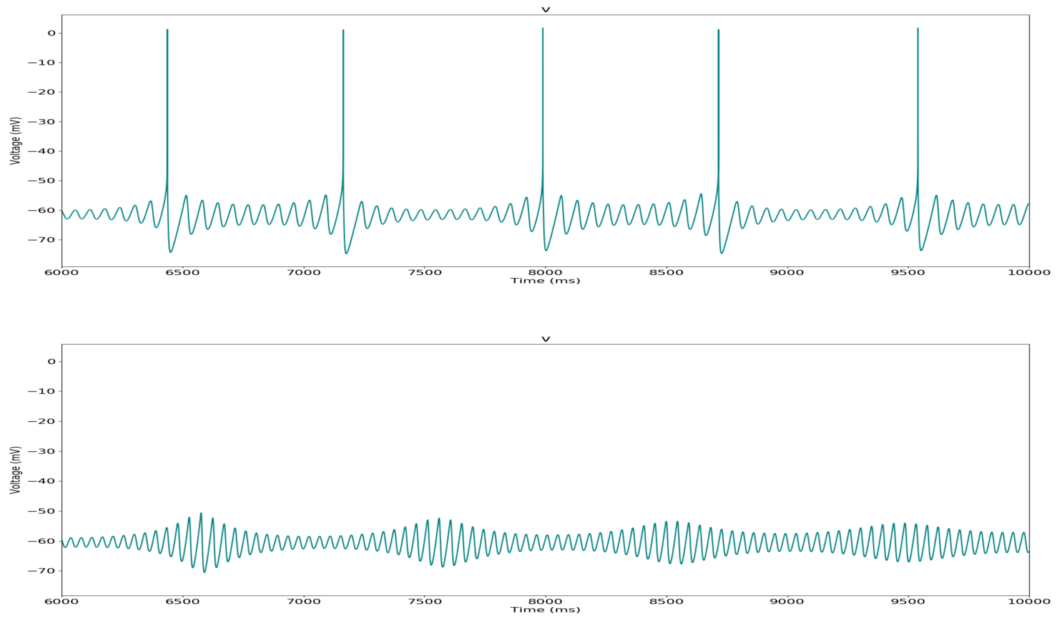


Figure 11: top: $\tau_m = 30$; bottom $= \tau_m = 15$

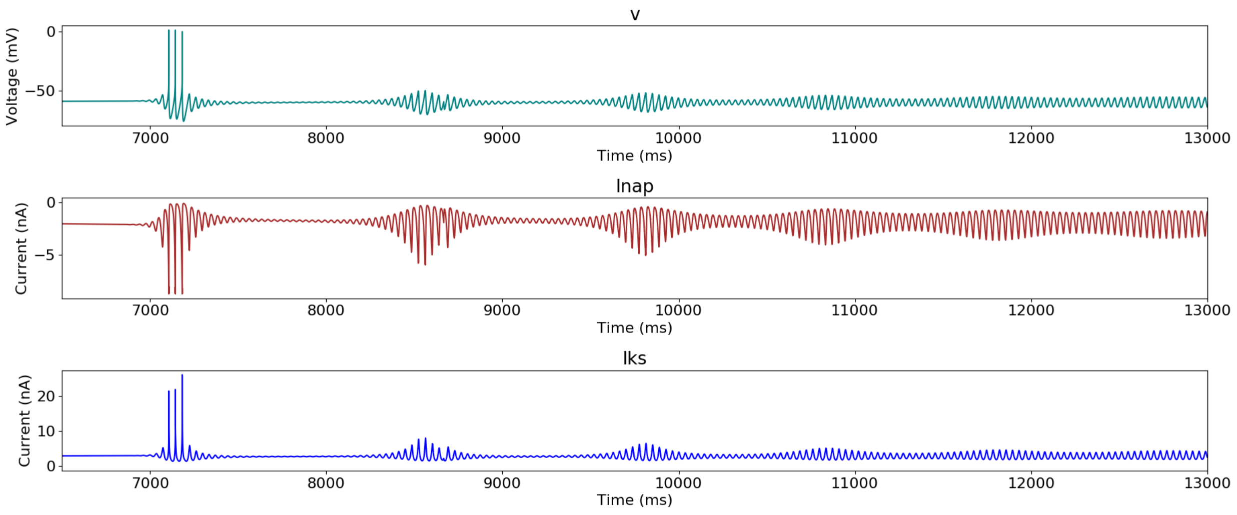


Figure 12: Channels I_{NaP}, I_{KS} effect

2.2.6 Channel model modules

Since silent living neurons do not always seem to have subthreshold oscillations, here we are going to present two more RTXI modules to be adapted to *Lymnaea* activity in order to generate such behaviour. For this purpose, based on RTXI module template (`https://github.com/RTXI/plugin-template`) and another available beta channel based on the module (`https://github.com/GNB-UAM/wang_1993_inap`).

First of all we are going to go through the most important methods into template files. From the different files available in the template, the ones that must be modified in order to create a new module are pluging-template.cpp and its corresponding .h, as well as the Makefile to build and install it.

In pluging-template.cpp, the main activity methods are available. Here is where all the settings are done in order to obtain a customised GUI. These methods could be divided in GUI setting and execution ones.

On the one hand, for GUI setting, parameters that are going to be available in the model must be defined, this is done in “DefaultGUIModel”, there, all variables are defined denoting: label, description and type. Label will be the variable identification for a later initialization or updating and in type we can select within the following ones:

1. OUTPUT, this will be handled by RTXI as an output value from the module, so it will appear in panels like connector panel or oscilloscope.
2. INPUT, analogous to OUTPUT, this parameter will handle data to the model as an input.
3. PARAMETER, parameter will be shown in the module panel and it is possible to change its value. When defining a parameter it is necessary to identify the type of it (DOUBLE, INT, etc).
4. STATE, this element will also be shown in module panel, however it will be also as a value reference, since it cannot be modified.

All these parameters are defined here and initialized in update method. In this flag-driven method, INIT, MODIFY, UNPAUSE, PAUSE and PERIOD actions are handled. Here, the most important ones for our module are INIT and MODIFY. Here, parameters and states, must be bounded to their C++ variable, relating a variable with a label. In MODIFY option, variable value is modified by GUI value.

In contrast to parameters and states which must be bounded with their corresponding variable, INPUT and OUTPUT types are accessed as lists, being located in the same order as they have been defined in “DefaultModeGUI”.

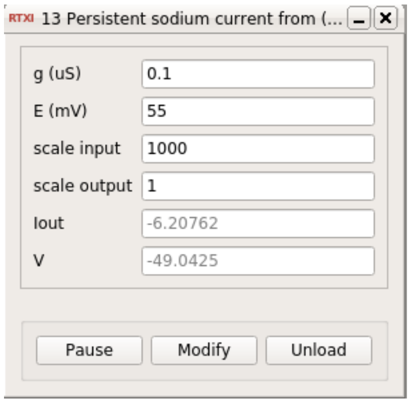
In addition, self-customised buttons can be added to the module. They can be added as a group into a “GUIwidget”, as it is specified in the template. An activity function can be bounded to this button, so it is executed anytime it is pressed.

On the other hand, the “execute” and “initParameters” methods handle scripts variables, independently from the GUI. In “initParameters”, variables are initialised as chosen, here GUI parameters can be modified as well as some auxiliary variables not showing in the GUI. On execute, code that will be execute each cycle script is added. RTXI runs as a loop, executing each module at each loop, these frequency is set in System Control panel and adapted to Real-Time performance.

After these description of the panels, we are going to get into detail of the channels modules.

Sodium Persistent Current (I_{NaP})

This channel available at `https://github.com/GNB-UAM/wang_1993_inap`, has a simple implementation, since the variables involved in this channel are only 3: g_{NaP}, V_{Na} and m_{∞}. The first two variables are defined as parameters, so its value can be adapted during the experiment to the characteristic of the living signal. The third parameter (m_{∞}) is a steady variable, which means, its value do not depend on its previous values, therefore it all can be computed in *execute* method, reading v from the input variable and writing it back into the output variable. In figure 13 there is a screenshot of this module.



g (uS)	0.1
E (mV)	55
scale input	1000
scale output	1
Iout	-6.20762
V	-49.0425

Figure 13: Persistent Sodium Channel Module

This open source module is going to be adapted to the experimental necessities, so it might be changed adding some GUI parameters or states.

Slowly Inactivation Potasium Current (I_{Ks})

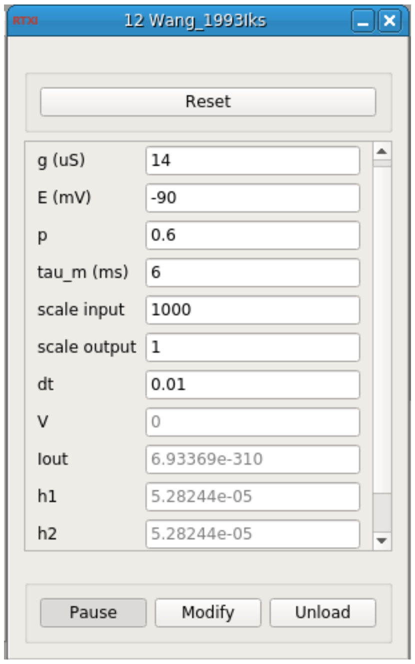
In combination with I_{NaP}, this channel generates the subthreshold oscillations. Since it is not enough with sodium current isolated, it is necessary to implement this other channel as a module in RTXI. For this purpose, RTXI template is going to be used as a reference to develop this module, following I_{NaP} module GUI style, so their combination can be more intuitive during hybrid experiments.

This current is more complex in terms of variables, so its GUI will also have more elements. Apart from the increase in variables number, some of them are dynamical, which supposes an additional issue. To handle dynamical variables, some variables not included in GUI are going to be defined, using the same code used to implement the model, with the corresponding functions, adapted to the module class as auxiliary methods.

Dynamical variables value depends on the time instant and, consequently, on the previous values. Therefore this will be taken into account when updating values in each iteration when the method execute is computed. All values will be updated using the equations described in Wang et al. [X.-J. Wang, 1993] and integrated using Euler method. For this purpose, there will be necessary to set a time step (dt). This time step will define the precission with which the model will be generated, i.e. the same model with 0.1 and 0.001 as dt will generate the same signal, but in the case of 0.001 much more iterations will be necessary obtain the same result as with 0.1.

In model terms, it is usually not important how precise the result is. However, when connecting this model to a living neuron, it must adapt to it (see section 1.4). Since RTXI do not perform automatic adaptation, a dt parameter will be added to GUI so it can be modified depending on the experiment necessities.

Therefore, the tunable parameters in this module will be the ones shown in figure 14. g and E are added as well as they were in sodium current, following the same nomenclature. They represent channel conductance and reversal potential. This last parameter will be necessary to be adapted regarding living signal amplitude, since it somehow is the threshold that determines whether the channel has or does not have effect.



g (uS)	14
E (mV)	-90
p	0.6
tau_m (ms)	6
scale input	1000
scale output	1
dt	0.01
V	0
Iout	6.93369e-310
h1	5.28244e-05
h2	5.28244e-05

Figure 14: Slowly Inactivation Potasium Channel Module

Another parameter used for this GUI will be tau_m, time activation constant for m. This parameter has been selected since, as it is related in Wang et al. [X.-J. Wang, 1993] controls subthreshold frequency, increasing as it increases its value.

While p value is also a constant for that equation, it will also be in the model, so it could be changed in case it is necessary.

In addition to the time step, there is another concept to take into account in adaptation terms. Value received from the living neuron is processed by the DAQ, and it might not be in the same scale. In order to make this module scalable and since it will not always receive action potential from the DAQ, there will be a scale parameter to tune this.

In the same way, value received by the amplifier, is usually interpreted as voltage and converted into nA, e.g. as 10nA = 1V. Therefore, analogous to the input scale, another parameter: *scale output*, will be included into the module.

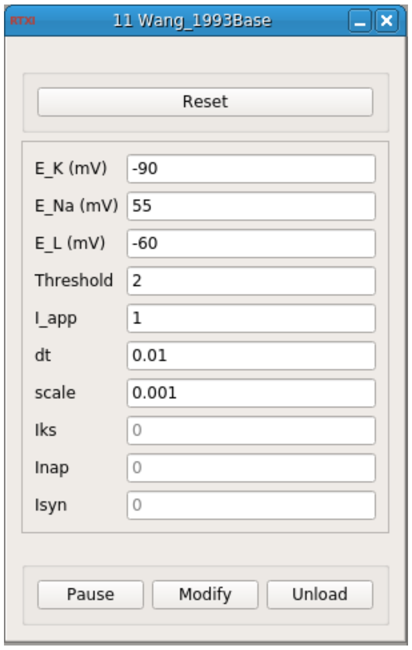
Regarding the dynamical variables presents in this model, there might be occasions during the experiment when it is necessary to reset all values. This is why an additional button will be added which will set all values to its default value.

Finally, some STATE values will be defined to monitor the values from variables such as the voltage or current generated and to follow their evolution. Furthermore, we will define the V value as INPUT and as the generated current value as OUTPUT.

Hodgkin-Huxley base model

As we have described in section 2.2.5, Wang et al. model [X.-J. Wang, 1993], defines two new channels whose interaction generates subthreshold oscillations over a Hodgkin Huxley Model, which is a basic model of integrate-and-fire type. Therefore, the rest of the C++ code is going to be adapted analogously to slowly inactivation potasium current module, in order to a possible test in RTXI of the two channels and their properties.

The figure 15 shows an screenshot of the module, with all its containing parameters.



E_K (mV)	-90
E_Na (mV)	55
E_L (mV)	-60
Threshold	2
I_app	1
dt	0.01
scale	0.001
Iks	0
Inap	0
Isyn	0

Figure 15: Parameters of the Hodgkin-Huxley base module following the model by [X.-J. Wang, 1993].

Since it has been analogous to previous model, we are not going to get into detail of the type of each element used, but an overview of the parameters in the GUI. E parameters are reversal potential of different ionic channels (I_K, I_{Na} and I_l), and threshold is σ value in Wang et al. [X.-J. Wang, 1993], used to adapt action potential activation in the model.

As it happened with I_{Ks} channel, this module has equations with dynamical values, which means they must be updated each iteration and they are dependent of dt integration value, as well as, its previous value. This is why, dt parameter is part of this module.

In slowly inactivation potasium channel, we added a scale parameter used to adapt input received to mV. Analogous to this issue, in this module, the input might not be in the adecuate scale either. This is why there is also a scale parameter in this module, so it can be easily adapted using RTXI.

Finally, analogous to potasium channel, a reset button will be included, so parameter reinit can be done.

3 Results

3.1 Validation of the experimental design in a conductance-based model

Using the model described in section 2.2.5 and a set of triplets implemented in C++ analogous to those implemented in the RTXI module, the triplets trial experiment has been computed over the computational model. The parameters of the model have been selected so that the model generates an intrinsic oscillation of a small frequency with no spike activation. The triplets setting has been selected to adapt to the oscillation frequency, with a small step. An example of this simulation is shown in figure 16.

The results of these simulations are presented in the form of ISI-maps, where ISI_1 is plotted against ISI_2, representing each point the presence of a spike generated as a response to the corresponding pulse (first, second or third). Since the model with oscillations is more likely to spike than a silent neuron, there might be spikes after all pulses (including first and second). Our interest is focussed on those that are produced after the third one.

To select synapse parameters, since the simulations are easier to be reproduced and it is possible to automatise them, several trials have been performed using different parameters to see which ones give better responses, regarding the goal of this experiment. The best parameters can be summarised as follows:

1. A_{syn} is related to the amplitude of the synapse. Values used for the mapping are from 60 to 75 with a step of 5. An example of its effect is found in figures 17 and 18, where A_{syn} values are 65 and 75 respectively, being more likely to obtain spikes after the pulses with higher A_{syn} value.
2. τ_{rec} do not really have much influence on the activation of trivial or non-trivial responses, small difference are appreciated under the same conditions with a difference of 100 times value. In figures 17 and 18 this is ilustrated with different values of A_{syn}, obtaining similar results in both even though τ_{rec} value goes from 50 to 750. In spite of the similarity, there are some differences, obtaining more activity when τ_{rec} is bottom.
3. τ_{fac} the effect of this parameter is illustrated in figure 19, where for the same condition for the other parameters, τ_{fac} has been changed from 50 to 200. The resulting map is not remarkably different, however it has some differencies, having more non-trivial results when τ_{fac} is 200.
4. τ_{in} has a big influence on spike activation facility, as bigger this parameter is, the most spikes result. This is shown in the map results of figure 20, since when the value is 14 it generates the third spike at almost any time.

Therefore, after analysing the figures here mentioned, it is found that the effect of the triplets into the model with intrinsic oscillations generates non-trivial responses, generating pulses when ISI_1 is smaller than half the period and ISI_2 is larger than this reference. We see that not only accelerating triplets and ISIs close enough are the only ones generating responses, but non preferred ISIs generate spiking response after the third pulse in the triplet.

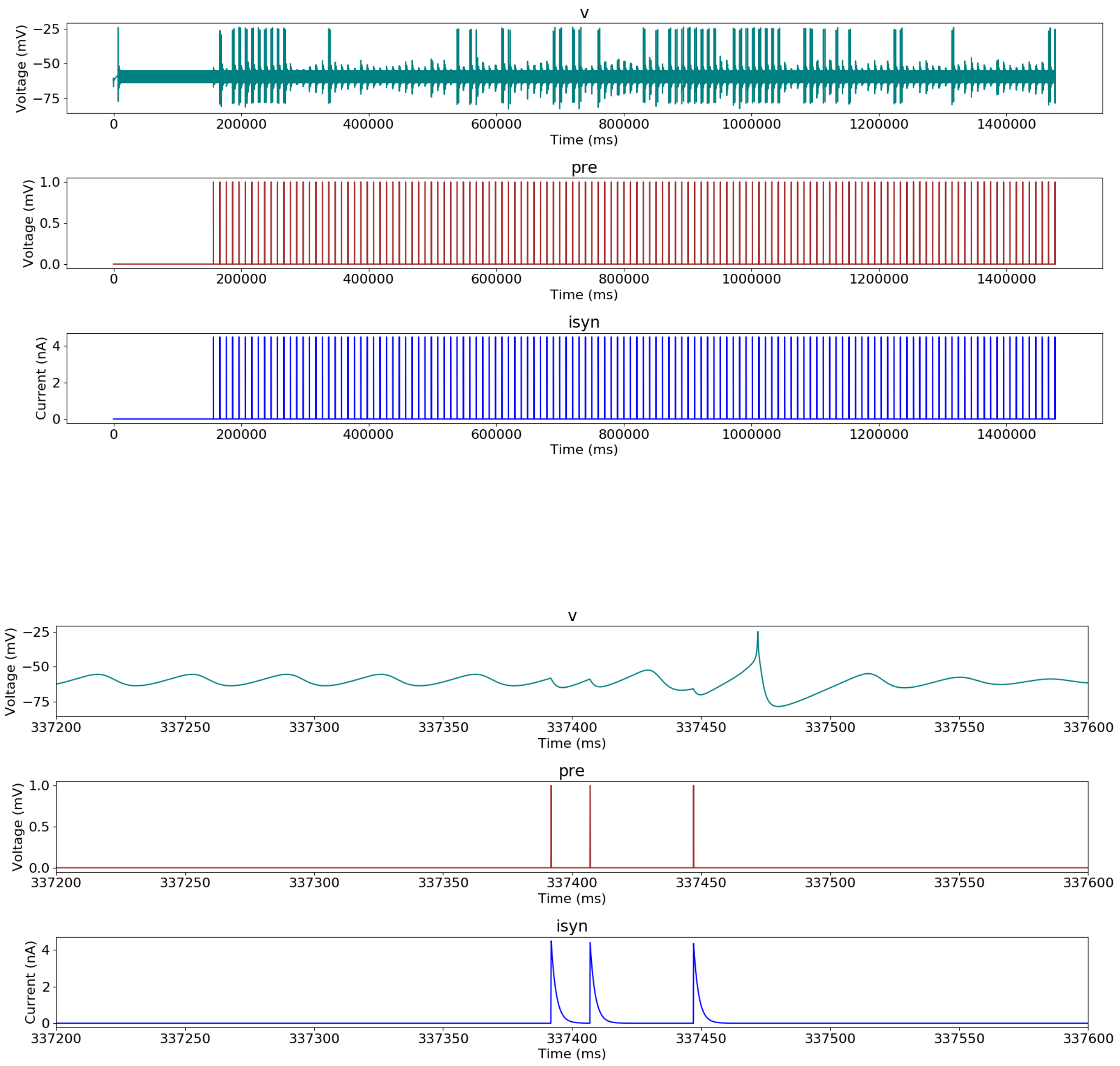


Figure 16: Simulation of Wang model in response to input triplets.

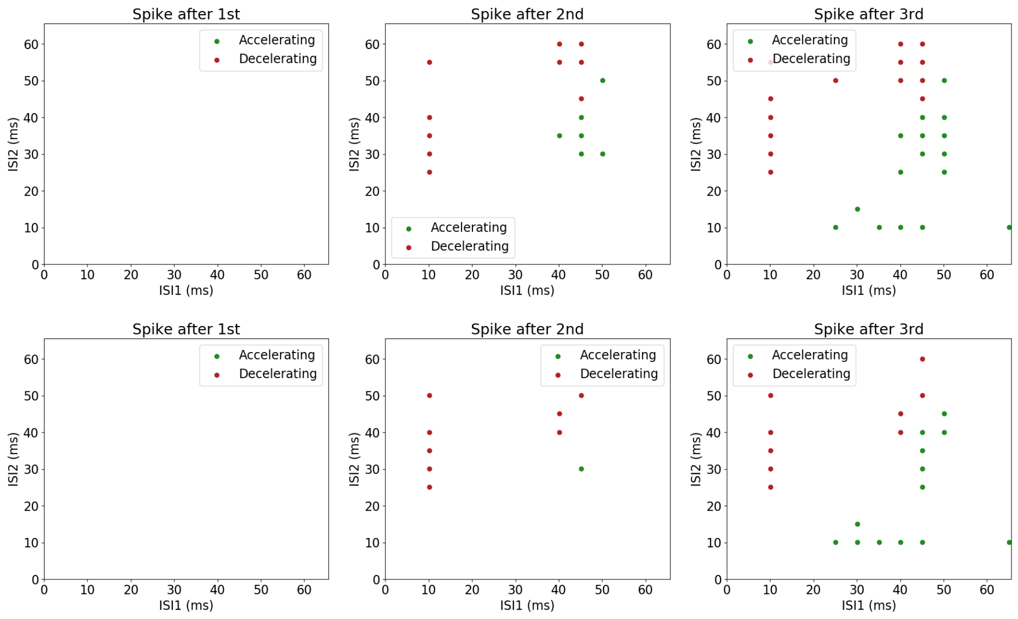


Figure 17: A_{syn}=65; $\tau_{fac} = 200$; $\tau_{in} = 2$; Top: $\tau_{rec} = 50$; Bottom: $\tau_{rec} = 750$;

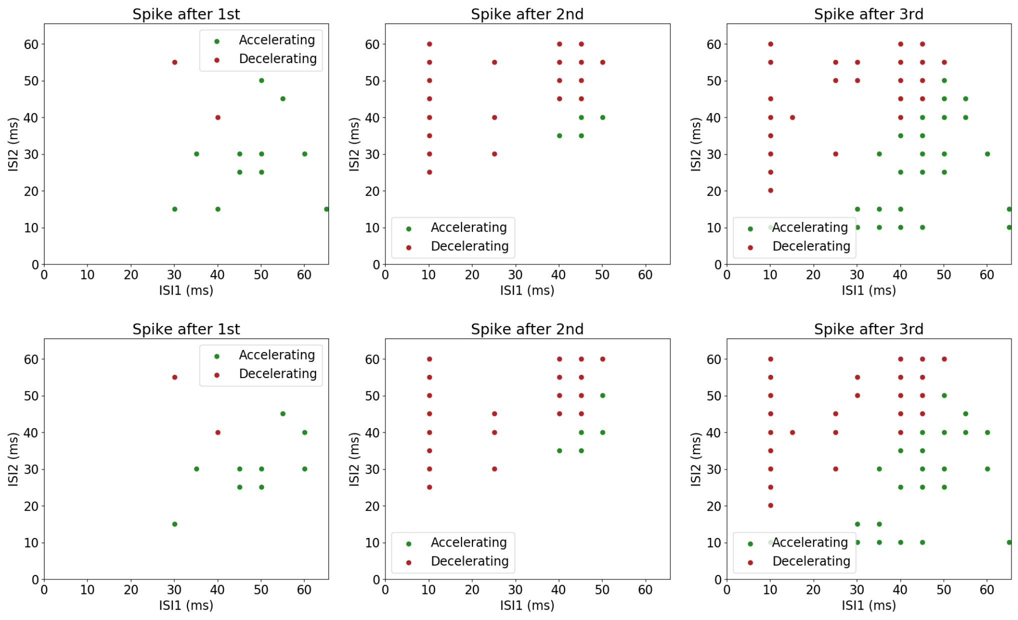


Figure 18: A_{syn}=75; $\tau_{fac} = 200$; $\tau_{in} = 2$; Top: $\tau_{rec} = 50$; Bottom: $\tau_{rec} = 750$;

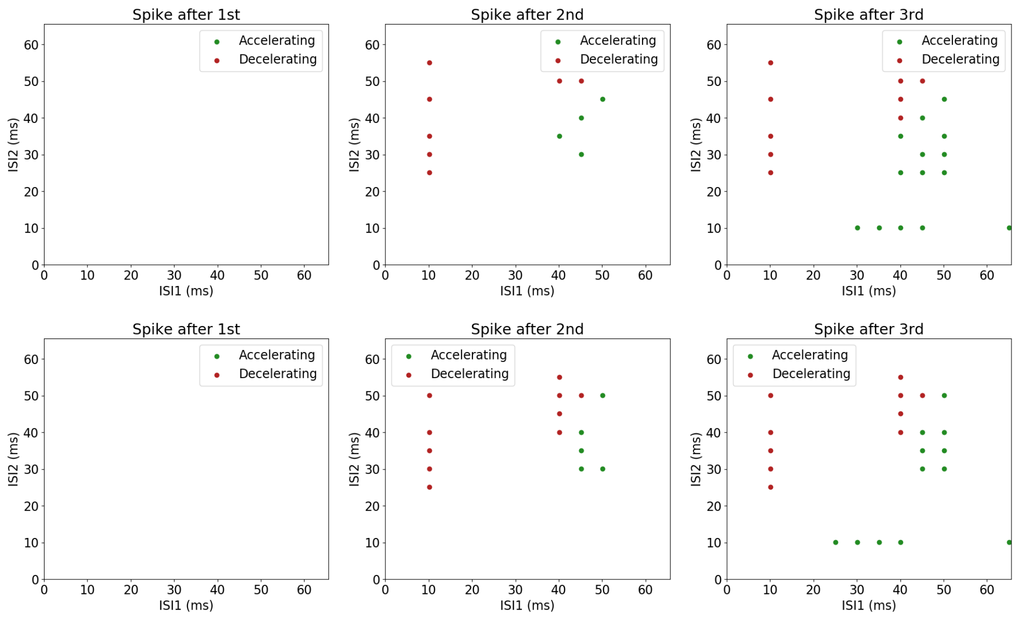


Figure 19: A_{syn}=65; $\tau_{rec} = 500$; $\tau_{in} = 2$; Top: $\tau_{fac} = 50$; Bottom: $\tau_{fac} = 200$;

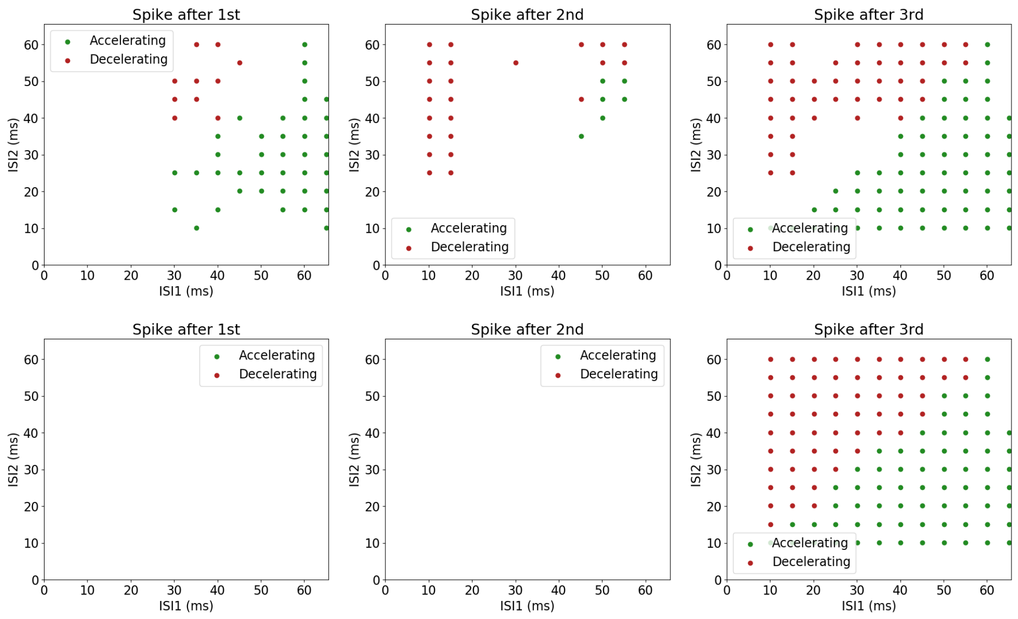


Figure 20: A_{syn}=65; $\tau_{rec} = 500$; $\tau_{fac} = 200$;Top:$\tau_{in} = 4$; Bottom:$\tau_{in} = 14$

3.2 Validation of experimental design in a living neuron

The aim of this experiment is testing these non-trivial responses into a living neuron. For a better understanding of the effect of these triplets of pulses into the neuron, here we are going to go through a few experiments with trivial responses.

3.2.1 Experiment on a quiescent neuron

For this purpose, a silent neuron in *Lymnaea*, with **no** subthreshold oscillation has been connected with pulses and synapse modules in RTXI. As a reminder, an accelerating pulse excites the neuron, generating a spike after the third pulse. In neurons with subthreshold oscillations the neuron is also more sensitive to excitation inputs, so it is easier that it produces pulses at any stimulus. In this case, as it is silent and has apparently no pre-synaptic inputs, is harder to make the neuron generate spikes.

On figure 21 there is an example of the effect of the pulses into a initially silent neuron. Pulses here have same amplitude, differences in plot amplitude are only for an easier differentiation of the three pulses.

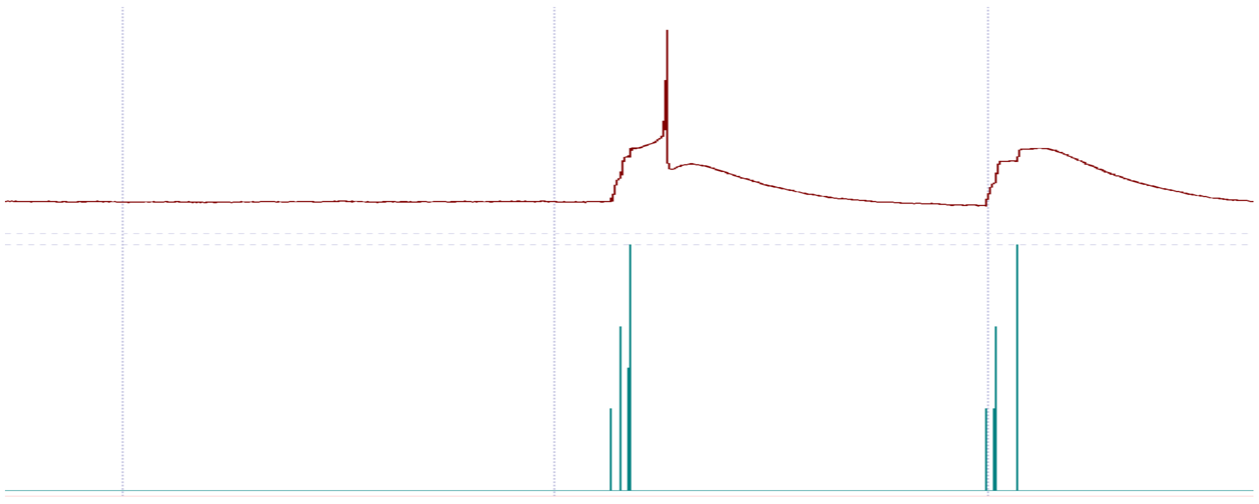


Figure 21: Example of pulses into initially silent neuron

After running the experiment into the living neuron and detecting the spikes, an ISI map is generated, plotting ISI_1 against ISI_2 and representing every time an spike occurs after the third pulse. The result is on figure 22. The code color represents as green points all spikes after an accelerating triplet and red points after a decelerating one.

As it can be appreciated in this result, most of the spikes activation have been generated by accelerating triplets, the trivial result expected, whereas the decelerating ones, only ends into a spike when ISIs are small enough.

This shows what is the real effect of these triplets on neural activity and how a triplet can alter it in a trivial way.

In a second trial, the same experiment protocol has been applied, but this time with a step of 0.2, so the result set is higher. The neuron selected for this experiment was silent but more active than the one used in the previous one. Furthermore, the preparation was used for neuron activity recording in the buccal ganglia for hours, and after that it was conserved in the fridge till the next day, when this experiment was performed. In the parietal ganglia most neurons where still active and responding to artificial stimulation using the amplifier.

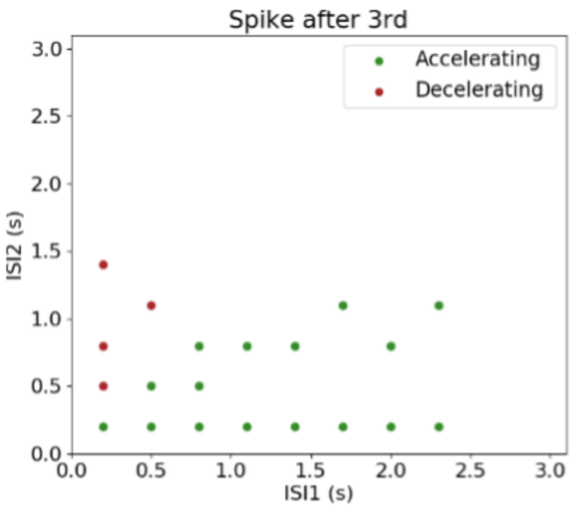


Figure 22: ISI map of trivial response experiment

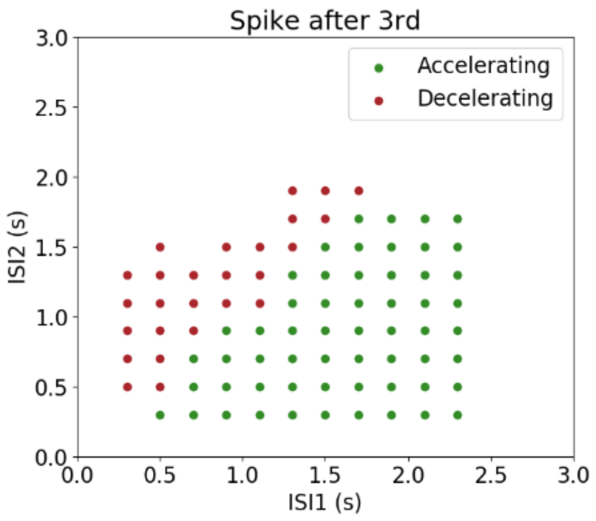


Figure 23: ISI map of trivial response experiment with step=0.2

The result of the experiment is shown in figure 23, where most pulses are a response to an accelerating triplet. The ones responding to a decelerating one are only the ones with ISIs small enough so the pulse can be generated.

Compared to the previous one, there are more spikes generated, this is due to the previous state of the neuron, since, as it was mentioned, it was more active than the other one.

3.2.2 Experiment on a neuron with subthreshold oscillations

Finding a neuron with intrinsic neuron oscillations is not something easy in the ganglia here explored. However, some of these neurons are connected and receive pre-synaptic impulses from each other, which generates a effect of subthreshold oscillations, even though they are not intrinsic of the neuron but from the pre-synaptic impulse received. In figure 24 there is an example of this kind of oscillations, there it can be appreciated that the oscillations has a fast increase and then a progressive decreasing, which is typical from **post-synaptic responses**.

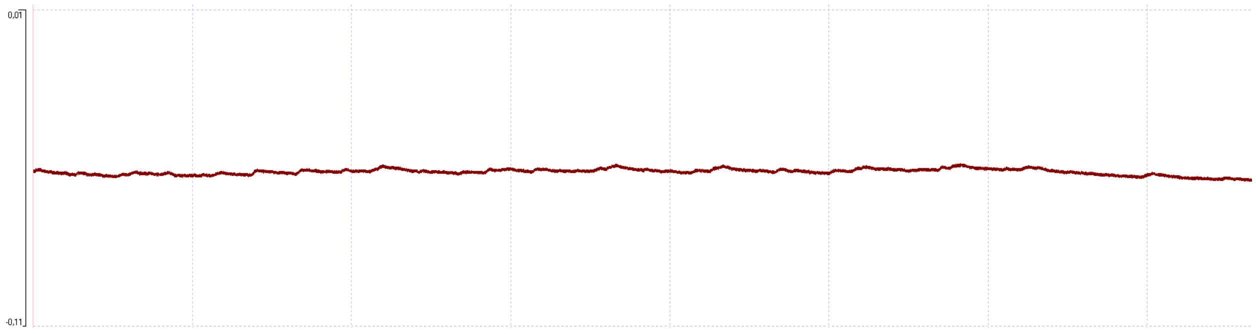


Figure 24: Subthreshold oscillations generated by pre-synaptic input example

Here in this example we are going to see the effect of these triplet pulses into a neuron of that kind. In figure 25 there is an example of the performance of this experiment. Since it is receiving inputs, it is much more active than a silent one, so it is easier to generate spikes. For this reason, we are going to see here maps not only for the activation of a spike after the third pulse but for all three spikes.

The resulting maps of two different trials under the same conditions and with small time lapse between them can be appreciated in figures 26 and 27.

In both maps several responses are found, mainly after second and third pulse. In fist trial (figure 26, there are some responses to decelerating triplets that could be classified as non-trivial, since ISI_1 is much bigger than ISI_2. However, seeing experiment 2 where this phenomena has not been reproduced again, motivates to think there might have been some activation of the neuron itself, independently from the neuron. Moreover, after spike two, most spikes are generated by deccelerating triplets, which could be caused by two reasons: non-trivial responses or the ISI values for the first two spikes is enough to make it spike. In that second case, the neuron would interpret the first two spikes as a stimulus by itself.

Hence, in this trial we have seen an example with oscillations generated by a synaptic input. Although it is not enough to prove the aim of this work, it is a good example of the effect of the triplets into a neuron with oscillations that could result in an easier activation than in the previous case.

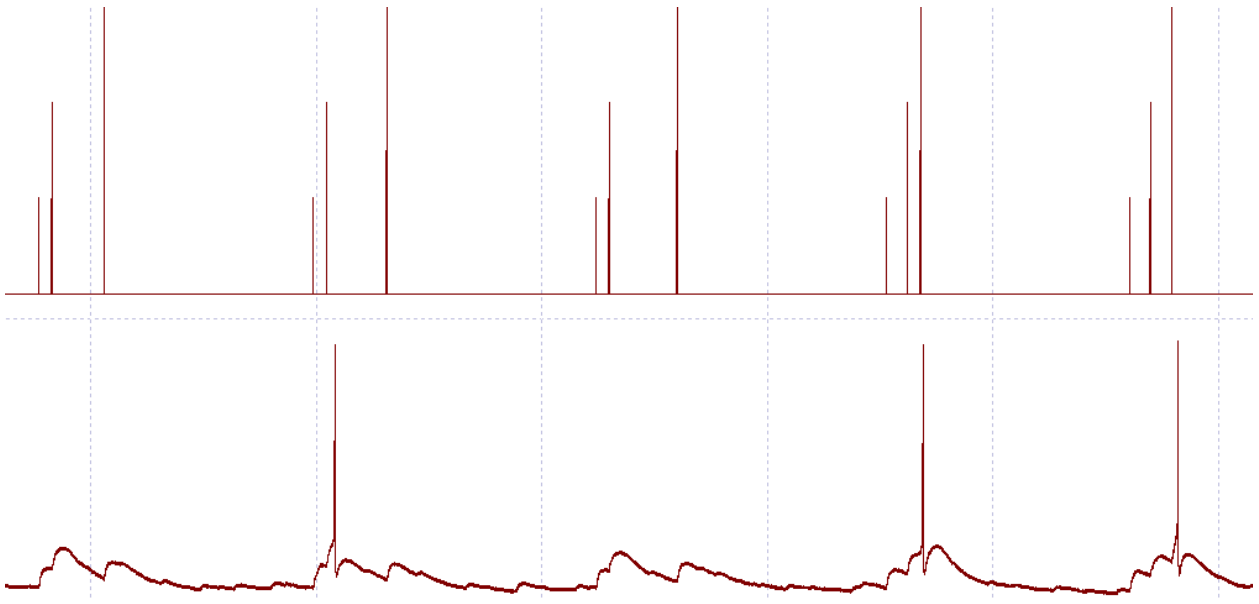


Figure 25: Example of the effect of triplets

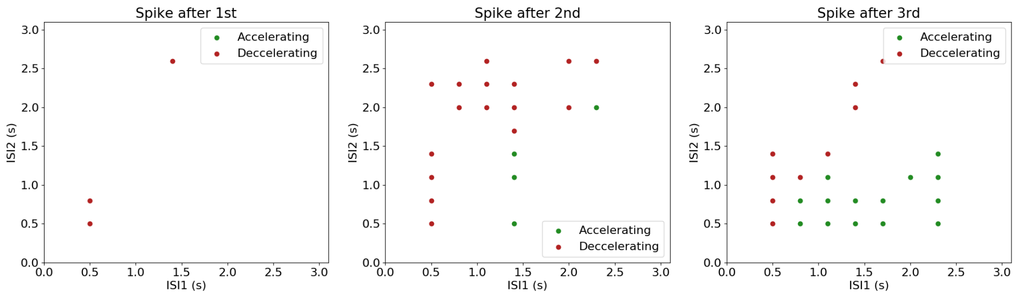


Figure 26: Resulting map for subthreshold pre-synaptic activity example. Trial 1.

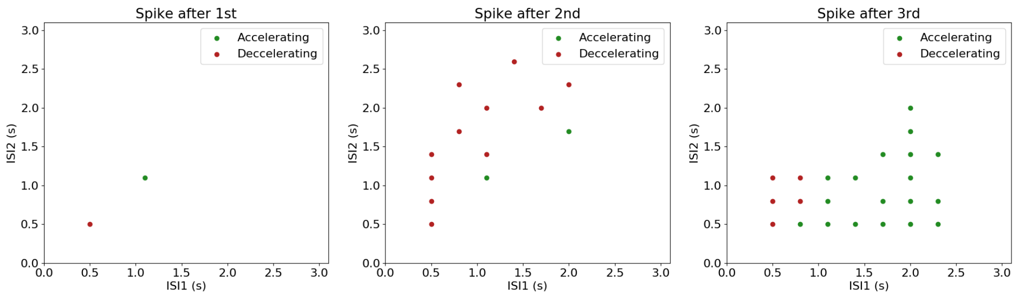


Figure 27: Resulting map for subthreshold pre-synaptic activity example. Trial 2.

3.3 Adding channels to induce subthreshold oscillations in a model. Proof of concept

In order to test module channels implemented and prove their value to generate subthreshold oscillations, first, they will be be tested in RTXI with a Hodgkin Huxley model, using leakage, sodium and potasium currents as defined in Wang et al. [X.-J. Wang, 1993].

Adaptation to computational model module in RTXI We saw in section 2.2.6 the implementation of a Hodgkin Huxley module, based on the currents from Wang et al. which will be reference from now on as "base module", this way, we have 3 modules, being I_{NaP} and I_{Ks} channels independent from the rest of the model.

To test their implementation and functionality in RTXI they have been all loaded into the program. Since all modules are independent, it is necessary to bound their inputs and outputs using the connector module, so they can act as a whole.

Pulses triplets over base module

As a first trial, we are going to prove the trivial behaviour using the base model isolated. This way, we have a Hodking Huxley classic neuron, connected to Tsodyks synapse, which is receiving output from triple pulse stimulator. This connection is illustrated in figure **??**.

Hodking Huxley neuron was tuned so it had no spontaneous spiking, and synapse parameters were also adapted so the current injected in the neuron was adequate. This way, the three modules were active generating each of their corresponding activities in real time, for the corresponding duration of the Triplets simulation.

The results for this trial are shown in map of figure 28. In this trial we have a result similar to the ones in section 3.2, most spikes are generated after an accelerating triplet and the ones caused by a decelerating one are only when ISIs are small enough.

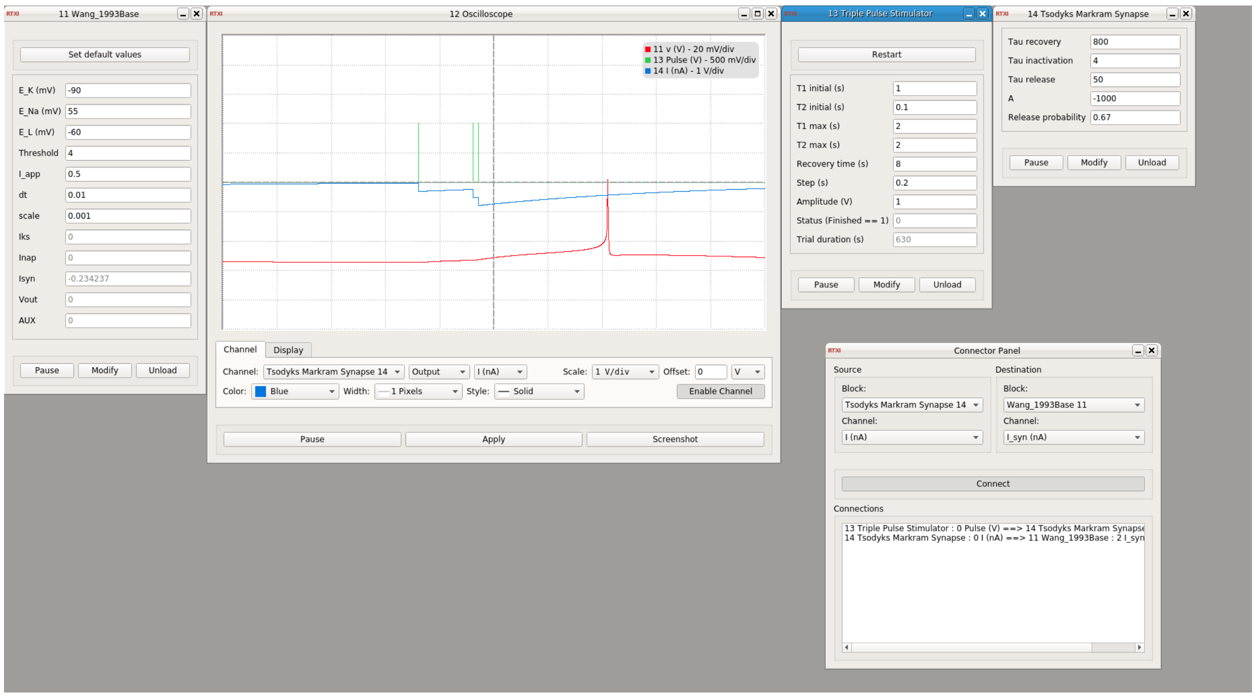
Therefore, here we have another prove of the experiment, but this time in a real-time enviroment, each module generating activity and interacting with each other as it would be done in a living neuron.

Integration of I_{KS} and I_{NaP} modules

Using the same set up of the previous trial, now modules corresponding to persistent sodium and slowly inactivation potassium channels are going to be loaded into RTXI to be connected to the base module.

Now, apart from the synapse current, the base module is going to receive also input from I_{KS} and I_{NaP} modules. Even when the experiment was run in a no-real-time scenario, it was costly in time to reach the subthreshold oscillations with no other external influence. Therefore, in real-time, this results in a considerably long experiment, where it could take even hours for the model to generate subthreshold oscillations by itself. For this reason, it might be necessary some initial spike generation, in order to initiate I_{KS} and I_{NaP} currents, and the subthreshold oscillation.

Taking all this into account, a real time experiment analogous to the previous one has been



E_K (mV) -90
E_Na (mV) 55
E_L (mV) -60
Threshold 4
I_app 0.5
dt 0.01
scale 0.001
Iks 0
Inap 0
Isyn -0.234237
Vout 0
AUX 0

Channel: Tsodyks Markram Synapse 14 Output I (nA) Scale: 1 V/div Offset: 0 V
Color: Blue Width: 1 Pixels Style: Solid

T1 initial (s) 1
T2 initial (s) 0.1
T1 max (s) 2
T2 max (s) 2
Recovery time (s) 8
Step (s) 0.2
Amplitude (V) 1
Status (Finished == 1) 0
Trial duration (s) 630

Tau recovery 800
Tau inactivation 4
Tau release 50
A -1000
Release probability 0.67

Source
Block: Tsodyks Markram Synapse 14
Channel: I (nA)
Destination
Block: Wang_1993Base 11
Channel: I_syn (nA)

Connections
```
13 Triple Pulse Stimulator : 0 Pulse (V) ==> 14 Tsodyks Markram Synapse
14 Tsodyks Markram Synapse : 0 I (nA) ==> 11 Wang_1993Base : 2 I_syn
```

Figure 28: RTXI screenshot of the test

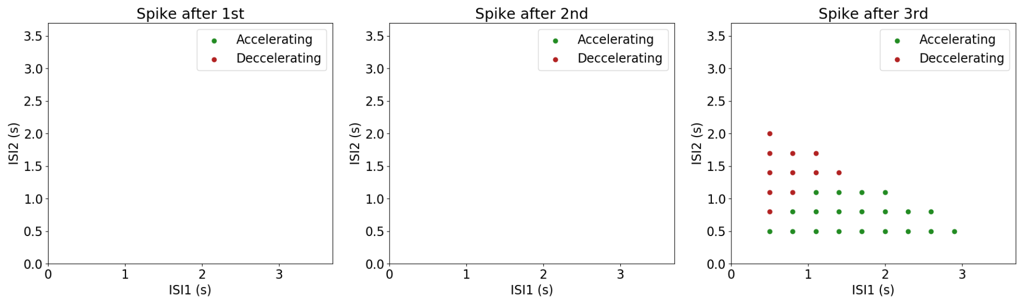
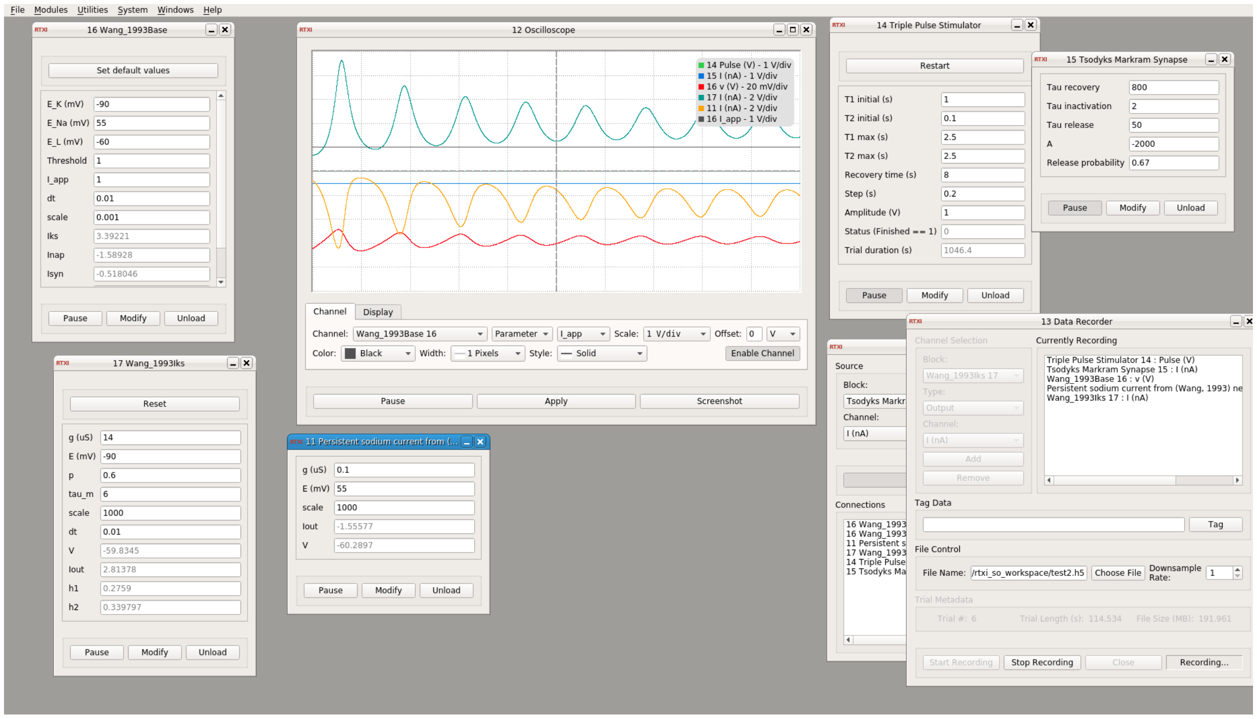


Figure 29: Resulting map for Hodgkin Huxley real-time RTXI example.

performed, but this time with the presence of subthreshold oscillations. An example of these modules is found in figure 30, being in the oscilloscope represented the subumbral oscillations achieved. In figure 31 there is an extract of the recordings obtained by this experiment performance. In that example, there are shown different non-trivial responses, since the triplet is a decelerating one and the neuron is producing an spike in response. Furthermore, it can be appreciated the effect of the synapse in the neuron, since the spike is produced at the highest value of the synapse.

After detecting the generated spikes and their relation with each pulse in the triplet, the resulting map is shown in figure 32. This time, there are found more responses to decelerating triplets, particularly, there can be appreciated some spikes generated when ISI_1 is much smaller than ISI_2



16 Wang_1993Base

E_K (mV)	-90
E_Na (mV)	55
E_L (mV)	-60
Threshold	1
I_app	1
dt	0.01
scale	0.001
Iks	3.39221
Inap	-1.58928
Isyn	-0.518046

17 Wang_1993Iks

g (uS)	14
E (mV)	-90
p	0.6
tau_m	6
scale	1000
dt	0.01
V	-59.8345
Iout	2.81378
h1	0.2759
h2	0.339797

12 Oscilloscope

- 🟩 14 Pulse (V) - 1 V/div
- 🟦 15 I (nA) - 1 V/div
- 🟥 16 v (V) - 20 mV/div
- 🟩 17 I (nA) - 2 V/div
- 🟧 11 I (nA) - 2 V/div
- ⬛ 16 I_app - 1 V/div

Channel: Wang_1993Base 16 Parameter I_app Scale: 1 V/div Offset: 0 V
Color: Black Width: 1 Pixels Style: Solid

11 Persistent sodium current from (...

g (uS)	0.1
E (mV)	55
scale	1000
Iout	-1.55577
V	-60.2897

14 Triple Pulse Stimulator

T1 initial (s)	1
T2 initial (s)	0.1
T1 max (s)	2.5
T2 max (s)	2.5
Recovery time (s)	8
Step (s)	0.2
Amplitude (V)	1
Status (Finished == 1)	0
Trial duration (s)	1046.4

15 Tsodyks Markram Synapse

Tau recovery	800
Tau inactivation	2
Tau release	50
A	-2000
Release probability	0.67

Source
Block: Tsodyks Markr
Channel: I (nA)

Connections
16 Wang_1993
16 Wang_1993
11 Persistent s
17 Wang_1993
14 Triple Pulse
15 Tsodyks Ma

13 Data Recorder

Channel Selection
Block: Wang_1993Iks 17
Type: Output
Channel: I (nA)

Currently Recording
Triple Pulse Stimulator 14 : Pulse (V)
Tsodyks Markram Synapse 15 : I (nA)
Wang_1993Base 16 : v (V)
Persistent sodium current from (Wang, 1993) ne
Wang_1993Iks 17 : I (nA)

Tag Data

File Control
File Name: /rtxi_so_workspace/test2.h5 Downsample Rate: 1

Trial Metadata
Trial #: 6 Trial Length (s): 114.534 File Size (MB): 191.961

Figure 30: RTXI screenshot of ionic channels modules test

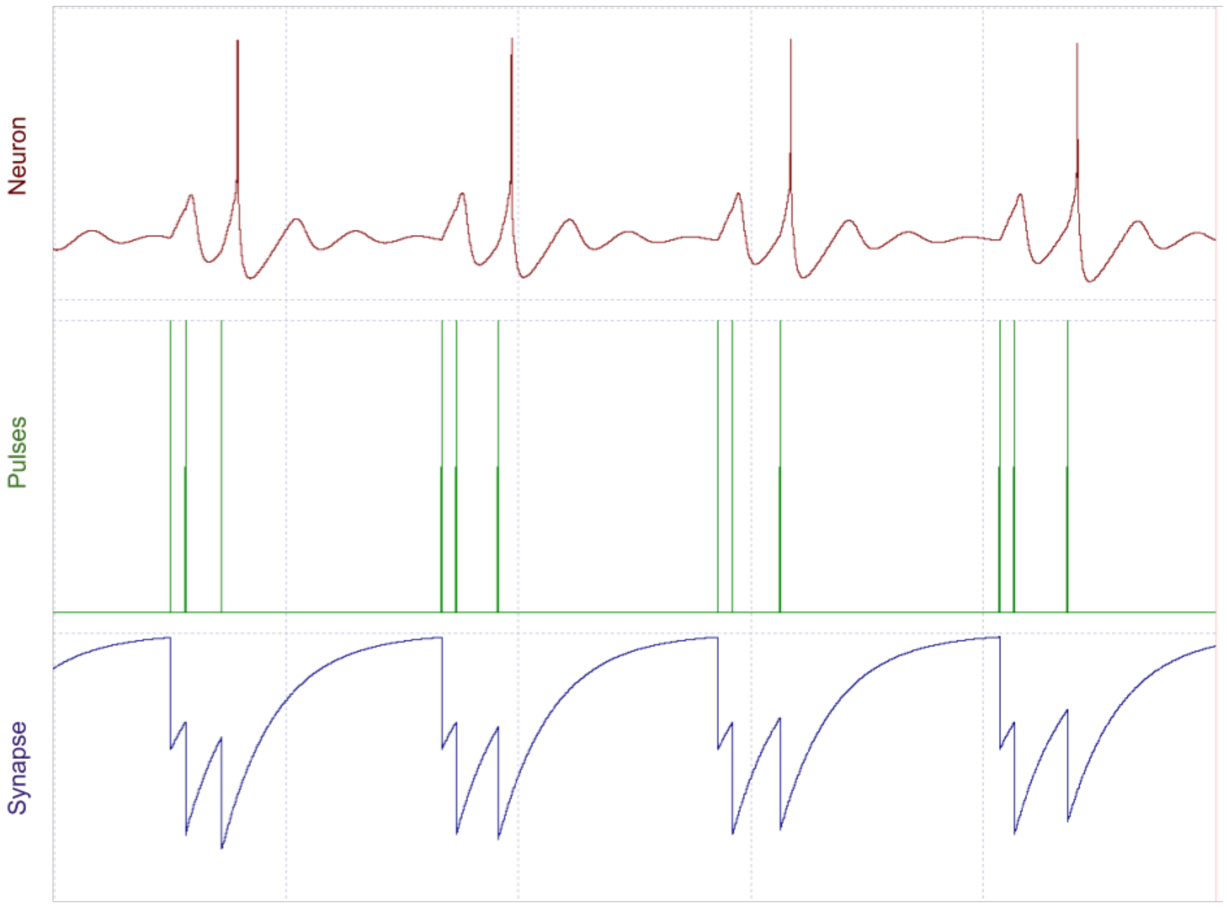


Figure 31: Experiment example of subthreshold oscillations. Neuron, pulses and synapse register (top, middle and bottom, respectively).

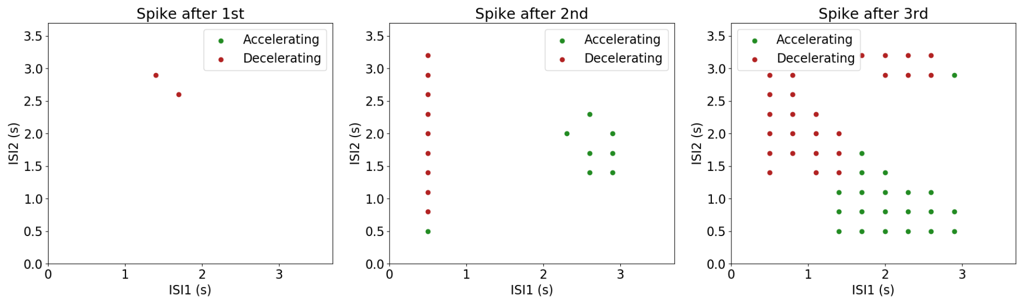


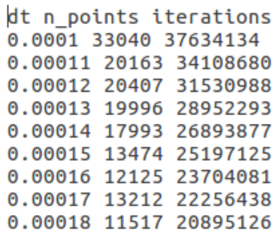
Figure 32: Resulting map for ionic channels integration real-time RTXI example.

3.3.1 Module adaptation for the implementation of hybrid experiments

As it has been mentioned before, adapting these channels to a living neuron is not something trivial. Since RTXI do not provide automatic adaptation of the model to the living signal, it is necessary to adapt it by hand. Thanks to the parameters defined in the module and the ones added to I_{NaP}, this can be done during the experiment.

On the one hand, as it was mentioned during this document, time scale is an essential element in artificial-living adaptation. In adaptation algorithms set by Reyes et al. [Reyes-Sanchez et al., 2018] to adapt this parameter, some event is taken as a reference. For the example adaptation they use the event selected is a burst, so a burst in the computational model, must be conformed by the same points as in living one. In this experiment, the issue is different, there are no burst implied, so in order to adapt it, we have chosen a spike as an event.

For an easier adaptation during the experiment, Wang et al. [X.-J. Wang, 1993] model has been executed with different values of integration time step (dt), concretely in the range of (0.0001, 0.001) with step 0.00001, and (0.001, 0.1001) with step 0.0001. The result of this has been stored into a reference file with the value of dt, the number of points of a spike and the number of iterations needed. In figure 33 there is an extract of this file.



```
dt n_points iterations
0.0001 33040 37634134
0.00011 20163 34108680
0.00012 20407 31530988
0.00013 19996 28952293
0.00014 17993 26893877
0.00015 13474 25197125
0.00016 12125 23704081
0.00017 13212 22256438
0.00018 11517 20895126
```

Figure 33: Header of spike length in points reference

Another crucial issue we have mentioned before is amplitude, however, in this case, Wang model has a amplitude of range 60, which is similar that the spike amplitude, *Lymnaea* usually has. Therefore, the adaptation in terms of voltage will only be necessary in the reversal potential, which during the experiment, may be adjusted changing the reversal potential parameter value in each current module.

Finally, there is one more thing to take into account when adapting these ionic currents to the living neurons. In the model, the current value is in terms of nA, this value is sent directly from RTXI to the DAQ board output, which is directly connected to the amplifier where the current will be injected to the neuron. However, the amplifier converts received input from voltage to nA, (since for it, every single value received is in voltage). The conversion performed is 10 nA for each 1V. Therefore, the output value generated by the modules in nA, will not be received in the amplifier with the same value, but as 10 times the original value. For this reason, currents output might be scaled to be 10 time less, so it can be received by the amplifier with the correct value.

4 Discussion

Throughout this project, we have addressed the design and validation of a hybrid circuit built for stimulating living *Lymnaea* neurons to test their response on artificially generated spike trains delivered by a model dynamic synapse.

The aim of the hybrid experiment designed in this work is to test the presence of non-trivial input/output responses that arise from the combined interactions of intrinsic subthreshold oscillations and dynamic synapses in a living neuron that receives spike trains with distinct temporal structure. For this purpose the experiment was implemented in a real-time open-source software platform, RTXI, using existing modules and implementing new ones to build an effective hybrid circuit.

On a first approach, the experiment was validated using a conductance-based model proposed by [X.-J. Wang, 1993]. This model is capable of generating subthreshold oscillations, thanks to the interaction between I_{NaP} and I_{Ks} currents. Therefore, a stimulation that explored different ISIs configurations was designed and applied to this model, using the Tsodysk-Markram dynamic synapse model as a mediator for delivering the stimulus [Tsodyks and Markram, 1997]. This protocol was implemented in C++, and its execution was automatised in order to test different synapse parameters and assess the characterisation of the different responses.

In this part of the project it was set out the effect of each synapse parameter in the experiment. Moreover, we validated in an additional model that it is possible to obtain non-trivial responses when subthreshold oscillations are present.

In a second validation of the hybrid experiment, the experiment was tested on a real living system using RTXI modules. For this purpose different neurons in parietal ganglia where selected to interact with the modules, receiving the synaptic current generated by RTXI. In these experiments, we considered three cases. In two of them, the neuron was quiescent and the resulting ISI-maps showed trivial responses, which means the triplets of pulses where able to activate the neuron only in the cases of accelerating triplets.

The third validation example presented was performed on a neuron which had subthreshold oscillations. However, due to their waveform, they seemed to be caused by a pre-synaptic input, not by intrinsic activity. Here the ISI-maps obtained presented more spikes where the response was non-trivial. However the results where not conclusive to indicate if this effects where due to the non-trivial response.

Since finding a neuron with subthreshold oscillations is not easy in this system, we implemented them artificially. In order to obtain a reproducible experiment, two modules were used to generate subthreshold activity into the living neuron. This two modules corresponded to the two ionic channels that induce subthreshold oscillations in the Wang et al. model [X.-J. Wang, 1993]. This way, I_{KS} was implemented in a module for RTXI, including the necessary parameters taking into account the necessary adaptations between the module and the living neuron during the experiment. Complementary to this module, the I_{NaP}

module, which is available in open source, was included in the experiment, adding some adaptation parameters analogously to the potassium module.

As a proof of concept of the combination of these modules with the others required for the hybrid circuit implementation, the experiment was performed in RTXI in real-time. For this purpose, a neuron base model was used first, based on the channels used by Wang et al. besides the I_{KS} and I_{NaP} channels.

Therefore, the experiment, was first run over the base module, configuring the neuron to be silent and the synapse parameters to be adequate for the spike generation. In this experiment we found trivial behaviour, obtaining responses to the third pulse only when the triplet was accelerating, as it happened with the hybrid experiment with the living neuron.

Consecutive to this test, by using the sodium and potassium modules, subthreshold activity was activated into the neuron. As it was mentioned before, reaching this subthreshold oscillations was expensive in time, being necessary to force a spike to initiate it. The result of applying the triplets to the neuron with subthreshold oscillations where exposed into ISI-maps, which showed non-trivial oscillations, just like it happened when the experiment was performed in non real-time (see section 3.1).

Finally, after this proof of concept in the three neurons, we performed the needed adaptation to use these modules in a living neuron, specifically in *Lymnaea*.

To conclude, here we have presented the design of a hybrid experiment along with its validation with models under real and non-real time environments. Due to the difficulty on finding subthreshold oscillations, we designed and tested a solution using RTXI modules to artificially generate this kind of activity into a living neuron.

As future work, these modules will be further tested in hybrid experiments to study the effect of the combination of subthreshold and synaptic dynamics in shaping input/output preferences in distinct experimental setups and neural systems.

References

[Amaducci et al., 2019] Amaducci, R., Reyes-Sanchez, M., Elices, I., Rodriguez, F. B., and Varona, P. (2019). RThybrid: A standardized and open-source real-time software model library for experimental neuroscience. *Frontiers in Neuroinformatics*, 13:1–14.

[Ambroise et al., 2017] Ambroise, M., Buccelli, S., Grassia, F., Pirog, A., Bornat, Y., Chiappalone, M., and Levi, T. (2017). Biomimetic neural network for modifying biological dynamics during hybrid experiments. *Artificial Life and Robotics*, 22(3):398–403.

[Arroyo et al., 2013] Arroyo, D., Chamorro, P., Amigó, J. M., Rodríguez, F. B., and Varona, P. (2013). Event detection, multimodality and non-stationarity: Ordinal patterns, a tool to rule them all? *The European Physical Journal Special Topics*, 222(2):457–472.

[Baroni et al., 2010] Baroni, F., Torres, J. J. J., and Varona, P. (2010). History-dependent excitability as a single-cell substrate of transient memory for information discrimination. *PLoS One*, 5(12):e15023.

[Baroni and Varona, 2007] Baroni, F. and Varona, P. (2007). Subthreshold oscillations and neuronal input-output relationships. *Neurocomputing*, 70(10–12):1611–1614.

[Baroni and Varona, 2010] Baroni, F. and Varona, P. (2010). Spike timing-dependent plasticity is affected by the interplay of intrinsic and network oscillations. *Journal of Physiology Paris*, 104(1-2):91–98.

[Chamorro et al., 2009] Chamorro, P., Levi, R., Rodriguez, F. B., Pinto, R. D., and Varona, P. (2009). Real-time activity-dependent drug microinjection. *BMC Neuroscience*, 10:P296.

[Chamorro et al., 2012] Chamorro, P., Muñiz, C., Levi, R., Arroyo, D., Rodríguez, F. B., and Varona, P. (2012). Generalization of the dynamic clamp concept in neurophysiology and behavior. *PLoS ONE*, 7(7):e40887.

[Frega et al., 2012] Frega, M., Pasquale, V., Tedesco, M., Marcoli, M., Contestabile, A., Nanni, M., Bonzano, L., Maura, G., and Chiappalone, M. (2012). Cortical cultures coupled to Micro-Electrode Arrays: A novel approach to perform in vitro excitotoxicity testing. *Neurotoxicology and Teratology*, 34(1):116–127.

[Garrido Peña, 2019] Garrido Peña, A. (2019). TFM Part I: Experimental and theoretical characterization of CPG activity in Lymnaea stagnalis.

[Hindmarsh and Rose, 1984] Hindmarsh, J. L. and Rose, R. M. (1984). A model of neuronal bursting using three coupled first order differential equations. *Proceedings of the Royal Society of London. Series B, Containing papers of a Biological character. Royal Society (Great Britain)*, 221(1222):87–102.

[Hodgkin and Huxley, 1952] Hodgkin, A. L. and Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of physiology*, 117(4):500–44.

[Hong and Khan, 2017] Hong, K.-S. and Khan, M. J. (2017). Hybrid Brain–Computer Interface Techniques for Improved Classification Accuracy and Increased Number of Commands: A Review. *Frontiers in Neurorobotics*, 11:35.

[Izhikevich, 2003] Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6):1569–1572.

[Komendantov and Kononenko, 1996] Komendantov, A. O. and Kononenko, N. I. (1996). Deterministic chaos in mathematical model of pacemaker activity in bursting neurons of snail, Helix pomatia. *Journal of Theoretical Biology*, 183(2):219–230.

[Konrad and Shanks, 2010] Konrad, P. and Shanks, T. (2010). Implantable brain computer interface: challenges to neurotechnology translation. *Neurobiology of disease*, 38(3):369–75.

[Latorre et al., 2013] Latorre, R., Aguirre, C., Rabinovich, M., and Varona, P. (2013). Transient dynamics and rhythm coordination of inferior olive spatio-temporal patterns. *Frontiers in Neural Circuits*, 7(SEP):138.

[Latorre et al., 2006] Latorre, R., Rodríguez, F. B., and Varona, P. (2006). Neural signatures: multiple coding in spiking-bursting cells. *Biological Cybernetics*, 95(2):169–183.

[Latorre et al., 2016] Latorre, R., Torres, J. J., and Varona, P. (2016). Interplay between Subthreshold Oscillations and Depressing Synapses in Single Neurons. *PLoS One*, 11(1):e0145830.

[Le Masson et al., 2002] Le Masson, G., Renaud-Le Masson, S., Debay, D., and Bal, T. (2002). Feedback inhibition controls spike transfer in hybrid thalamic circuits. *Nature*, 417(6891):854–858.

[Levi et al., 2018] Levi, T., Bonifazi, P., Massobrio, P., and Chiappalone, M. (2018). Editorial: Closed-loop systems for next-generation neuroprostheses.

[Liu, 2000] Liu, J. W. S. (2000). *Real-time Systems*. Prentice Hall, Upper Saddle River, New Jersey.

[Llinas, 1988] Llinas, R. R. (1988). The intrinsic electrophysiological properties of mammalian neurons: insights into central nervous system function. *Science*, 242(4886):1654–1664.

[Mishchenko et al., 2018] Mishchenko, M. A., Gerasimova, S. A., Lebedeva, A. V., Lepekhina, L. S., Pisarchik, A. N., and Kazantsev, V. B. (2018). Optoelectronic system for brain neuronal network stimulation. *PLOS ONE*, 13(6):e0198396.

[Muniz et al., 2008] Muniz, C., Levi, R., Benkrid, M., Rodriguez, F. B., and Varona, P. (2008). Real-time control of stepper motors for mechano-sensory stimulation. *Journal of Neuroscience Methods*, 172(1):105–111.

[Nishikawa et al., 2019] Nishikawa, S. M., Khoyratee, F., Eiler, A.-C., Kim, S. H., Ihida, S., Toshiyoshi, H., Tixier-Mita, A., Ikeuchi, Y., Kohno, T., Aihara, K., Fujii, T., and Levi, T. (2019). Neuro-hybrid systems: the future for biomedical applications. *Seisan Kenkyu*, 71(4):787–790.

[Norman et al., 2016] Norman, S. E., Butera, R. J., and Canavier, C. C. (2016). Stochastic slowly adapting ionic currents may provide a decorrelation mechanism for neural oscillators by causing wander in the intrinsic period. *Journal of Neurophysiology*, 116(3):1189–1198.

[Nowotny and Varona, 2015] Nowotny, T. and Varona, P. (2015). Dynamic Clamp Technique. In Jaeger, D. and Jung, R., editors, *Encyclopedia of Computational Neuroscience*, pages 1048–1051. Springer New York.

[Patel et al., 2017] Patel, Y. A., George, A., Dorval, A. D., White, J. A., Christini, D. J., and Butera, R. J. (2017). Hard real-time closed-loop electrophysiology with the Real-Time eXperiment Interface (RTXI). *PLOS Computational Biology*, 13(5):e1005430.

[Pinto et al., 2000] Pinto, R., Varona, P., Volkovskii, A., Szücs, A., Abarbanel, H., and Rabinovich, M. (2000). Synchronous behavior of two coupled electronic neurons. *Physical Review E*, 62(2):2644–2656.

[Potter et al., 2014] Potter, S. M., El Hady, A., and Fetz, E. E. (2014). Closed-Loop Neuroscience and Neuroengineering. *Frontiers in Neural Circuits*, 8(115).

[Reyes-Sanchez et al., 2018] Reyes-Sanchez, M., Amaducci, R., Elices, I., Rodriguez, F. B., and Varona, P. (2018). Automatic adaptation of model neurons and connections to build hybrid circuits with living networks. *bioRxiv*, page 419622.

[Robinson and Kawai, 1993] Robinson, H. P. and Kawai, N. (1993). Injection of digitally synthesized synaptic conductance transients to measure the integrative properties of neurons. *Journal of Neuroscience Methods*, 49:157.

[Sharp et al., 1993] Sharp, A. A., O'Neil, M. B., L, A., and Marder, E. (1993). The dynamic damp: artificial condudances in biological neurons. *Tins*, 16:389–394.

[Szücs et al., 2000] Szücs, A., Varona, P., Volkovskii, A., Abarbanel, H., Rabinovich, M., and Selverston, A. (2000). Interacting Biological and Electronic Neurons Generate Realistic Oscillatory Rhythms. *Neuroreport*, 11(3):563–569.

[Torres et al., 2019] Torres, J. J., Baroni, F., Latorre, R., and Varona, P. (2019). Temporal discrimination from the interaction between dynamic synapses and intrinsic subthreshold oscillations. *BioRxiv*, pages 1–25, doi: 10.1101/727735.

[Torres and Varona, 2012] Torres, J. J. and Varona, P. (2012). Modeling Biological Neural Networks. In *Handbook of Natural Computing*, volume 1-4, pages 533–564. Springer, Berlin, Heidelberg.

[Tsodyks and Markram, 1997] Tsodyks, M. and Markram, H. (1997). The neural code between neocortical pyramidal neurons depends on neurotransmitter release probability. *P. Natl. Acad. Sci. USA*, 94:719–723.

[Varona et al., 2002] Varona, P., Aguirre, C., Torres, J., Rabinovich, M., Abarbanel, H., and Rabinovich, M. (2002). Spatio-temporal patterns of network activity in the inferior olive. *Neurocomputing*, 44-46:685–690.

[Varona et al., 2016] Varona, P., Arroyo, D., Rodríguez, F. B., and Nowotny, T. (2016). Online event detection requirements in closed-loop neuroscience. In El Hady, A., editor, *Closed-Loop Neuroscience*, pages 81–91. Academic Press.

[Varona et al., 2001] Varona, P., Torres, J. J., Abarbanel, H., Rabinovich, M., and Elson, R. (2001). Dynamics of two electrically coupled chaotic neurons: experimental observations and model analysis. *Biological Cybernetics*, 84(2):91–101.

[Vavoulis et al., 2007] Vavoulis, D. V., Nikitin, E. S., Feng, J., Benjamin, P. R., and Kemenes, G. (2007). Computational model of a modulatory cell type in the feeding network of the snail, Lymnaea stagnalis. *BMC Neuroscience*, 8(S2).

[X.-J. Wang, 1993] X.-J. Wang (1993). Ionic basis for intrinsic oscillations. *NeuroReport*, 5:221–224.

[Yarom, 1991] Yarom, Y. (1991). Rhythmogenesis in a hybrid system-interconnecting an olivary neuron to an analog network of coupled oscillators. *Neuroscience*, 44(2):263–275.