Models and Implementations of Timing Processes Using Artificial Life Techniques

Maciej Komosinski Adam Kups

Technical Report RA–05/09

Institute of Computing Science
Poznan University of Technology

November 2009

Abstract

This work presents the implementation of the Scalar Timing Model idea (STM) in a specific artificial neural networks environment, and examines the general algorithmic complexity and possible architectures for the task of selecting the longest stimulus from a sequence of stimuli. The STM is a popular and commonly used model that concerns perception of time intervals in studies on humans and animals. Many experiments are conducted nowadays in order to verify and study parameters and attributes of the STM. One of the goals of this work has been to implement this theoretical model in the setting of the artificial neural networks. The implementation demonstrated the need of some extra components in order to maintain functional adequacy of the model. Another goal was to study the influence of pulse generation on the representation of time within the implementation (i.e., the pacemaker-accumulator part). No additional mechanisms have been included that provide the scalar property, so the implementation follows the typical behavior of the clock-counter model. Experiments have been described that illustrate the process of comparison of lengths of two stimuli by an artificial neural network.

Contents

1 Introduction

The main motivation of this research was to design and study one of the possible implementations of the Scalar Timing Model (which is derived from the Scalar Expectancy Theory – SET) [28] using a simulation (artificial life) environment. The idea behind this approach is that this model is relatively well established [30] by many works in psychology, and, therefore, it is very interesting to see whether it can endure limitations of artificial neural networks. If it can, then the Scalar Timing Model (STM) is one step further on the way to become an adequate scientific construct describing and explaining mechanisms of time perception in human brain and *a fortiori* in human mind – which is its main goal from the beginning. On the other hand, implementing any theoretical model in some environment gives insight into its validity according to the rules of the employed environment. Therefore such implementation is a good test whether the model is constructed well.

The general advantage of the STM is its popularity among psychologists. There are many works including experimental data about effects and details of human time perception, trying to link observations with this theoretical construct. Obviously, the basis of this popularity is not a matter of taste, but rather a history of development of psychology of time and the explanatory power of the SET (for discussion about theories, see [31, 7]). Another important fact is that the STM is universal in the sense that it was first applied in studies of animal perception of time [28]. This origin gives hope that it concerns general and common phenomena. The STM is relatively well developed, with many verification trials performed so far, which is important because of at least two reasons:

- huge packages of data (collected during experiments) are very helpful when it comes to checking whether some particular implementation is adequate,
- scientific importance of this model justifies the need of the multidisciplinary work in developing it.

Last but not least, it is important that the structure and cognitive psychology conceptualization frame of the STM makes it relatively easy to implement in an Artificial Life environment.

The STM is not the only model popular among psychologists – for a review, see [7]. There exist theoretical solutions which provide timing mechanisms without employing the pacemaker [26], which is a special oscillatory mechanism (see the next section for more details). In this work we focus on the implementation of the STM, but if there is a need to use another model without the pacemaker in the future, the implementation can be changed easily in order to satisfy this demand.

An effective implementation of the STM may give answers to a few heavyweight questions. First of all, it can be tested whether a particular model is consistent with its own assumptions. This can be considered easy, but one should keep in mind that every validation process in such a case is done according to some set of environmental rules. These rules are more or less explicitly included in a simulation, where the model is implemented. In the particular case of this work, these are rules of signal transmission between artificial neurons. The signal processing rules are a frame according to which the model is validated.

The limitations of the accepted set of rules could be an advantage, but also a disadvantage. It could be a disadvantage when the goal is the translation of the model into the artificial neural network to bring it closer to the neurobiological level of explanation: the set of rules according to which connected artificial neurons work is quite different from the set of rules for biological neurons. On the other hand, this can be an advantage when the goal is to check whether the model guarantees effective processing of information (in this case, time information) in terms of a more general and logically founded principles.

Whether constraints of the implementation environment are an advantage or not depends on the scope of analysis. The STM can be adopted to some simulation environment, where elementary processing units (artificial neurons) can be interpreted as higher-level equivalents of biological units. This functional equivalence is justified as long as the key properties of biological units exist in the simulation; the problem of equivalence is pertinent to any theoretical model in its conceptual form that is being implemented. Once the implementation of the STM is validated, it will also be possible to decompose elementary units of implementation into lower-level processing elements. This operation should make it possible to ultimately reach the basic biological level of information processing.

The second important advantage of implementations is that they may allow to find interesting properties of the model. This is also one of the advantages of employing the Artificial Life approach. In this research, we used the Framsticks simulator [16] (see [20, 19] for other neuroevolutionary experiments that use Artificial Life techniques). The short list of benefits is presented below.

- Implementation and simulation of the model must follow constraints of the set of rules included in environment. It is possible that the theoretical model in its original form does not provide all the mechanisms needed to effectively process information. If that is the case, by making the model more concrete (by implementing), one can possibly find new hypothetical sources of variance of the effects of psychological mechanisms. Later, one can try to verify hypotheses derived from the model by preparing appropriate experiments.

- It is possible to verify not only the hypotheses concerning psychological approach to processes of time perception, but also concerning the biological basis of these processes.

- It is possible to find more than biological equivalents of timing processes. Given experimental data collected from humans, and a theoretical frame that abstracts regularities of this data, and given knowledge about human brain, it would be most interesting to explore the field between the psychological and the biological level of analysis, and find how one of these levels is determined by another. There are some references between the STM and the biological level of explanation (see pages 178–186 in [22]) which make our experiments a suitable framework for such considerations.

- Artificial Life environments are usually flexible, and thus the rules of simulation can always be adjusted to meet theoretical demands. These environments give the possibility to see *on-line* how the model performs. The Framsticks platform can support analysis of the time perception model in many ways.

- The implementation of psychological (or neuropsychological) time perception mechanism can be referenced to another model of a different process. This reference does not have to (and probably it should not) be restricted to comparing complete modules. It is possible to match particular elements of these models and gain information about more global processes.

- Another way to use the simulation environment is to work on improvements of the STM. This could be done manually by programming every interesting issue. There is, however, another option. Artificial Life environments and the Framsticks platform in particular, apart from supporting connectionist (neural) models, implement evolutionary algorithms (EAs). These algorithms can be used in a number ways – for example, they can be used to find optimized parameters of the whole implementation of the STM. Another way of using EAs is to find a more effective (adequate) model which can consist of elements of the STM, but can also find new mechanisms which work in the same manner as the STM, or in a different (possibly better) way.

- Finally, an Artificial Life platform can be used to include the STM considered here in an artificially designed creature, and to see how this model is actually used by the creature in a virtual world. The conditions under which the model is useful can be identified, and methods of evaluation of the model under particular conditions can be suggested or refined.

There have been earlier attempts to implement timing processes [29] – for example, an implementation of a transformed STM which was tested by Wearden and Doherty. In fact, this was a connectionist implementation of the Scalar Timing Theory developed by

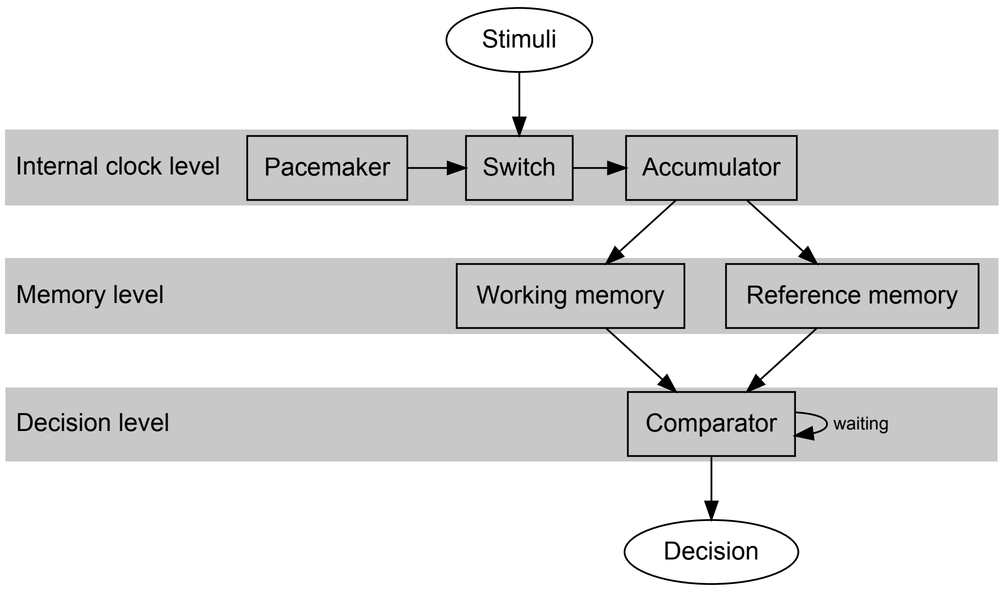

Figure 1: The general architecture of the Scalar Timing Model (STM).

Church and Broadbent. The difference between their simulation and our implementation is that the structure of their model on the conceptual level was concerned [29]

> ...representing an intrinsically dynamic process such as elapsing time in a fundamentally static network structure.

In our work, the conceptual level of the STM is preserved – for example, the pacemaker is represented by a single oscillator, whereas in the connectionist version it was represented by a set of oscillators. There is no simple way to arbitrate which solution is better – it should be verified by experiments. The strategy demonstrated in this paper is to stay as close as possible to the original concept of the STM. For further developments of our model, see [15].

2 Timing

2.1 The Scalar Timing Model

The Scalar Timing Model (illustrated on Fig. 1) derived from the Scalar Timing Theory contains three main parts [28]. First, the so-called internal clock level, consists of the pacemaker generating pulses and sending them to the accumulator which stores these pulses. The views on the importance of the pacemaker differ in various works [2, 26, 28]. Then, there is a switch which is considered to control the number of pulses stored in the accumulator as it is triggered by the stimulus. According to the assumptions of this model, the amount of accumulated pulses is a representation of the stimulus duration. Once the representation is acquired, it is in the next step sent to the second level of the model. This one is called the memory level and consists of two components – the working memory and the reference memory. Actually, time representations from the accumulator could be sent to both of these components, depending on the timing task. Simply said, the function of the reference memory is to provide useful information to compare with the content of the working memory[1] (which is in this case the time representation from the accumulator). The comparison process takes place on the third level of the model – the comparator. The results of the work of the comparator are the basis for further actions of the animal.

According to the literature [2, 1, 28], there are some more or less precise rules that describe the way these components work. The rules refer to the distribution form of some constants related to the threshold of the decision rule within the comparator component, the multiplication of information before it reaches the working memory, and the switch opening/closing latency. These rules are very important because they refer to the scalar and non-scalar sources of variability of judgements. The constants are not directly present in the implementation described here, since the main goal was to map the architectural scheme of the STM into the artificial neural network (ANN). However, the mechanisms of the ANN allow for emergence of regularities that correspond to some of the constants. These constants can also be manually added to the design of the ANN.

For the coefficient of variation to be truly constant (a property still not fully confirmed by psychological experiments [7, 27, 25]), the characteristics of the pacemaker should vary [26, 28], the accumulated value should be multiplied by a random variable [6], or other steps should be taken that concern the other components of the STM [6, 21].

The STM has two fundamental formal properties [28]:

- The mean representation of the stimulus duration should approximate the real stimulus duration (the *mean accuracy*).
- The coefficient of variation should remain constant independently from the stimulus duration. This property is similar to the Weber's law (the *scalar property*).

There are cases when these properties are not satisfied [27, 23], and this issue will be discussed later. The second property can be explained in terms of the working/reference memory mechanism and decision thresholds, as well as by other views that ascribe this property to the pacemaker mechanism [5, 6, 26]. However, the pacemaker mechanism was shown to be problematic to explain this property because of the primary assumption regarding pulse distribution (*Ibid.*). Here we will show that the assumption about pulse distribution of the pacemaker can provide some interesting data.

The experimental data collected across several experiments revealed that the scalar properties are often not obeyed. Thus, many sources of variance are reported of which some are non-scalar (see for example experiments designed to explain sensory-like sources in terms of the STM [30]). These sources of variance should be categorized and explained in order to understand the processes that underlie human time perception.

The main goal of the present implementation was to create a realization of the STM model that will be capable of comparing durations of the two subsequently appearing stimuli. It is a simple implementation which does not take into consideration numerous theoretical as well as experimental facts, and all the possible nuances of the time perception mechanisms. However, the comparison of two stimuli appearing one after another is a relatively frequent procedure (for a brief description of the research timing procedures and their application, see [24, 9]). Comparing two stimuli across several trials set up in some special manner (see for example [12]) is a psychophysical procedure used to discover the difference thresholds (DTs) in humans. DTs provide useful information

[1]See [8] for more information on problems with temporal memory.

about temporal resolution of the timing mechanisms. It is noteworthy that different structures of stimuli influence the DTs. Understanding the mechanisms underlying the process of simple comparison of two stimuli may form a basis for understanding of more complex processes.

What is important in the STM is that its original architecture is only a general idea about how time perception works. Every implicit assumption about processing of the two stimuli can be accepted or discarded according to the set of rules embedded in a specific simulation environment. In our implementation, the functional modules of the neural network reflect the components of the STM.

2.2 Algorithmic analysis of the selection and sorting tasks

While we are primarily concerned with the situation where a subject is asked to tell which of the two stimuli is longer, this situation is a special case of a more general task:

- There are n stimuli $(s_1, s_2, .., s_n)$,
- The task is to rank (sort) them according to their property (e.g., length).

For $n = 2$, the sorting task is equivalent to the task of indicating which stimulus is longer. The most efficient implementation (i.e., the one that needs only one "register" or a memory cell) is illustrated on Fig. 2. The first stimulus adds to the register, while the second one deducts from it. The resulting value in the register (called the accumulator here) contains quantitative information about the relative length of both stimuli. It can be further compared to 0 to decide which stimulus was longer, or express indecision if the magnitude of the accumulated value is too small.

More complex architectures for $n = 2$ are obviously possible, and they are necessary for $n > 2$, where one accumulator does not suffice. Note that in practice,

1. It is much easier to compare a pair of stimuli at once, than more of them all at one time. Therefore when comparing $n > 2$ stimuli, we often resort to comparing pairs.
2. In this regard, it is helpful if
   - the considered property of the stimuli is transitive (e.g., we seek the longest, shortest, lowest frequency, nicest, ...),
   - and the perception of this property is objective and deterministic.

   This allows to reduce the number of pairwise comparisons: since $s_i > s_j \wedge s_j > s_k \Rightarrow s_i > s_k$, a limited number of pairwise comparisons can yield a partial or a total order over all stimuli[2].

Assuming that properties mentioned above hold, for $n > 2$ stimuli and the sorting task, we generally need n "memory cells". As consecutive stimuli are perceived, they need to be compared with the representations of previous stimuli in memory, and sorted. We could call this behavior "on-line sorting". "Off-line sorting" would consist in first

[2]Preferences generally don't have to be transitive. If they are not, the number of comparisons cannot be reduced easily. In such cases, it may also be impossible to strictly sort stimuli or tell which single one is the extremum.

memorizing all stimuli and then sorting them, which might be more difficult for a human. In any case, we need many comparisons to sort n stimuli: at least $n \log n$, and $\frac{n(n-1)}{2}$ (all pairs) in the naive approach.

For $n > 2$ stimuli and the selection task (finding the extremum stimulus), we could use the sorting approach, but there exists a more efficient solution. While stimuli are presented, one only needs to keep track of the longest stimulus perceived so far, and compare each new stimulus with the one held in a single "memory cell" (this is the "on-line selection" approach). Therefore the selection task for $n > 2$ resembles the $n = 2$ case, and for n stimuli we only need n comparisons as outlined below:

1. A stimulus is presented.
2. Is this the first stimulus?
   (a) Yes: memorize its length – it is the longest one for now.
   (b) No: if its length is higher than the longest one, consider the current one the longest and memorize it.
3. Wait for the next stimulus; go to 1.

The condition (2) can be simplified by ensuring that the memory cell contains the representation of stimulus magnitude that is smaller from any magnitude that can actually be perceived; such a smallest magnitude could be "zero" and then the procedure would be:

1. Ensure that the memory cell stores "zero".
2. A stimulus is presented.
3. If its length is higher than the longest one, consider the current one the longest and memorize it.
4. Wait for the next stimulus; go to 2.

This section focused on the simplest possible approaches known in engineering. It is unknown whether corresponding approaches are present in animal brains – more sophisticated, redundant architectures may be involved, and conscious and unconscious functioning may differ. However, describing related technical solutions and specifying them as shown above may help in experimentation with natural, neural structures, their understanding and validation of the models.

2.3 Comparison of two stimuli: simple models

This work is primarily concerned with implementations of the comparison mechanism of $n = 2$ stimuli. As shown in the previous section, it requires a single accumulator cell. The main idea of the comparator is shown on Fig. 2. Fig. 3 assumes that we have a single sensor, where signal ≤ 0 means "no stimulus", and a positive input value means perceived stimulus. This flowchart demonstrates how a gap between stimuli is properly interpreted (when the first stimulus ends, the Accumulator stops increasing, we move to the right loop and start decreasing the Accumulator once the second stimulus begins).

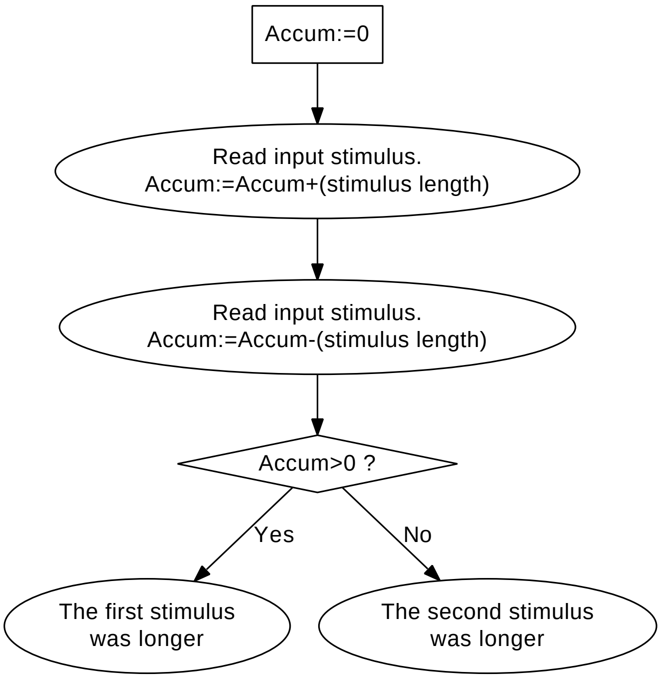
Accum:=0
Read input stimulus.
Accum:=Accum+(stimulus length)
Read input stimulus.
Accum:=Accum-(stimulus length)
Accum>0 ?
Yes
No
The first stimulus was longer
The second stimulus was longer

Figure 2: The general flowchart of the possible implementation of the timing model. Note that there should actually be three branches (if Accum=0, then the two stimuli are of equal length). We discard this option for simplicity.

The gray “Pacemaker/delay” box is optional and can be read as the “wait for some time” (“wait for the next pacemaker pulse”) unit. If present, the Accumulator will be modified less often which will result in lower precision of the accumulated value. It may also introduce some kind of bias depending on if, and how, the delay varies. The following sections present experiments with non-deterministic delays that show the influence of non-determinism on the accumulated values; see [14] for related theoretical investigations.

In the previous architectures, we have assumed that the Accumulator is capable of counting pulses up and down, i.e., it can add and subtract. If this is not the case and the Accumulator can only add stimuli, there is a need for another Accumulator (and the ability to compare both accumulated values), or some other entity (e.g., memory) that can deal with the comparison [11]. Fig. 4 shows a sample flowchart where a WM (Working Memory) cell is used to store the accumulated value of the first stimulus, and then deduct, or compare to, the second stimulus.

3 Implementation of the Scalar Timing Model

3.1 Simulation environment

For the implementation and experiments with timing, we have chosen the Framsticks software system [16, 17] because of the following:

- It allows to model and simulate networks of units that process signals,

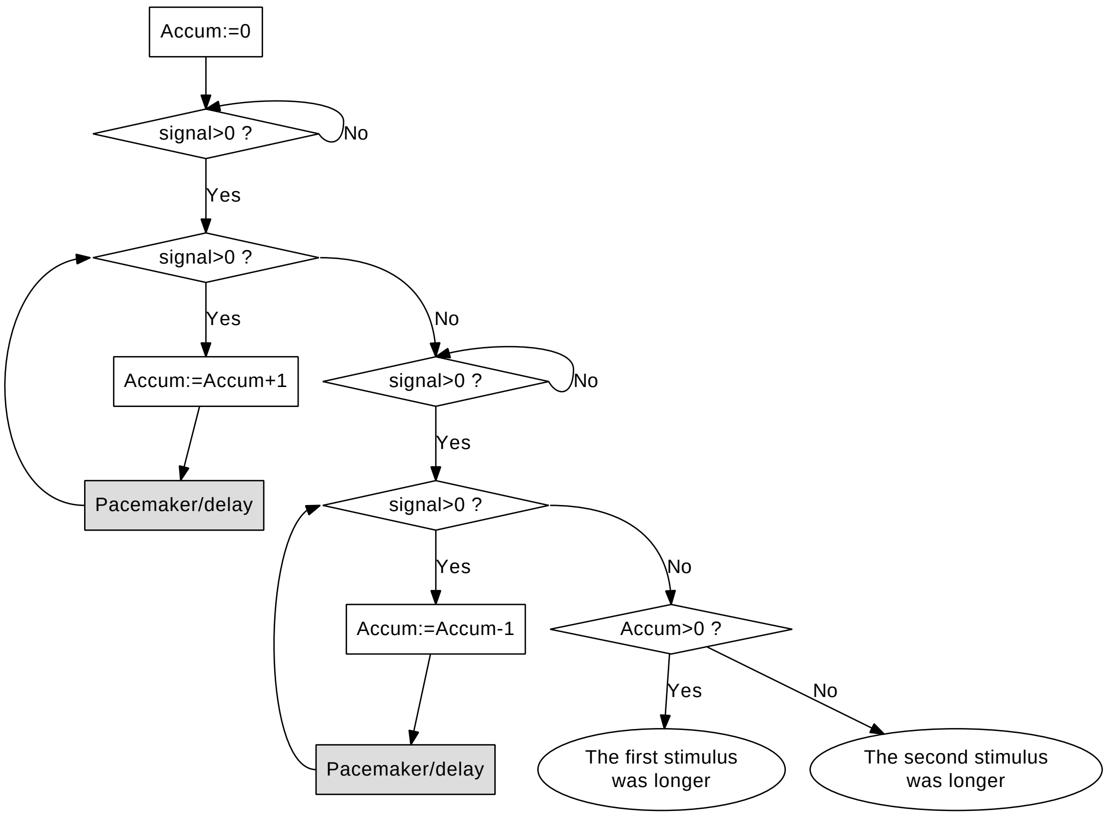

Figure 3: The flowchart of the timing procedure demonstrating detection of stimuli (periods where signal > 0 vs. signal ≤ 0).

- New types of signal-processing – or neural – units can be easily designed, modified and tested,
- The networks can be *embodied* and *situated*, i.e., they can be used to control virtual creatures or constructs,
- They can also be evolved and this process can be controlled in a number of ways,
- The simulator is powerful and flexible, requires little work to design diverse experiments,
- There is a graphical application available that facilitates interactive experimentation,
- There is a console application suitable for automated or computationally intensive processes,
- It can be easily integrated with other software (for further data analyses, charts, etc.),
- It is available for all major platforms and operating systems,
- It is supported and continuously developed.

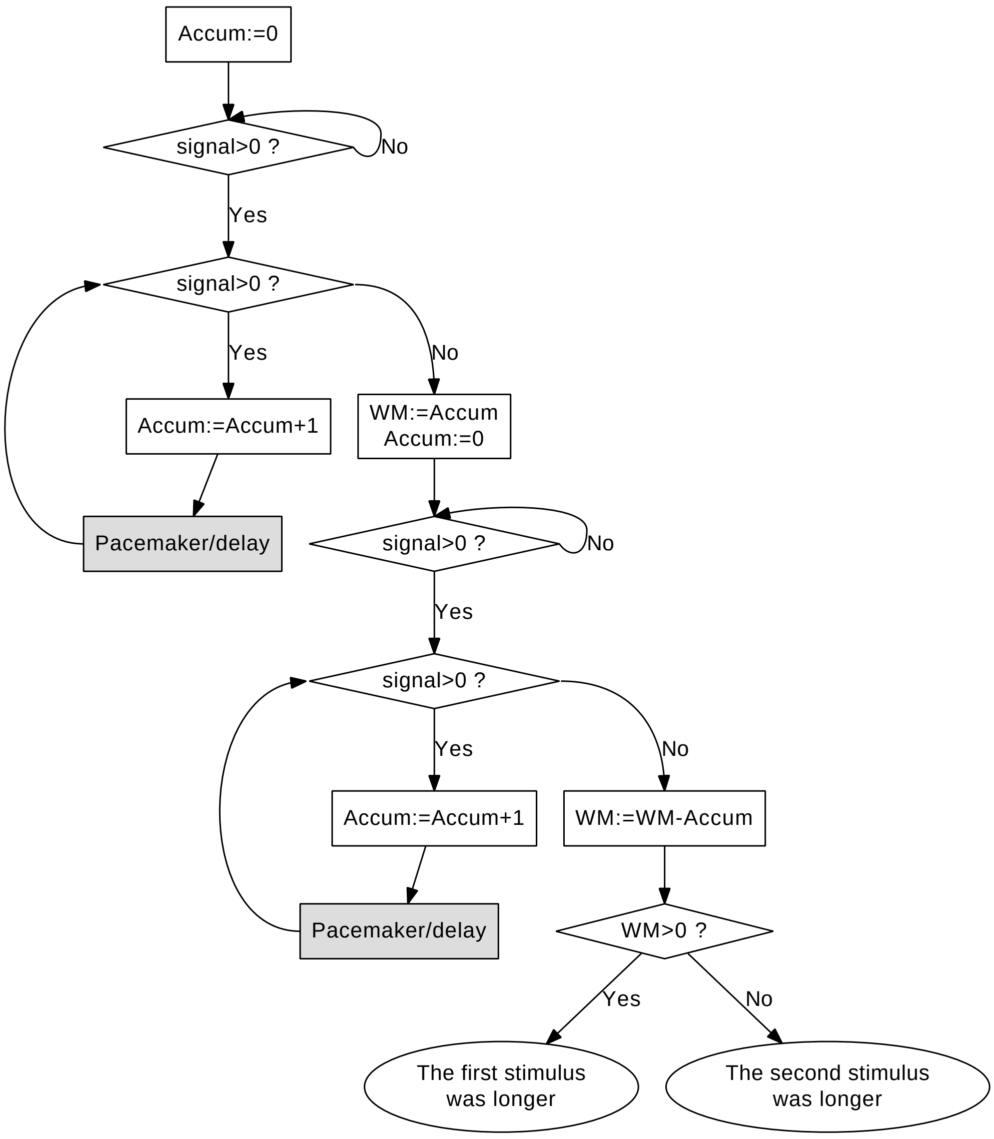

Figure 4: The flowchart of the timing procedure with the Accumulator only adding pulses, and the Working Memory (WM) responsible for comparing accumulated values for both stimuli. Note that the subtraction (WM:=WM−Accum) can be avoided: if possible, the contents of the WM (accumulated value for the first stimulus) could be directly compared to Accum (accumulated value for the second stimulus).

3.2 Implementation considerations

In its general form, the implemented model (presented in Fig. 6) consists of modules that reflect parts of the STM. There are, however, some issues that should be pointed out.

1. In the implementation, we use continuous (analogue) values of the neural signals (while computer floating point values are in fact discrete, they can be considered continuous for the purpose of these experiments). The model does not require that the signals are discrete or continuous, with the exception of the pacemaker.

2. To pass information in the neural network we use raw signal values (i.e., the neurons are not spiking, and the information is not encoded as frequency). Spiking neurons would resemble the way biological neurons work, but would also add an undesired layer of complexity.

3. In the STM, the rules of information processing in the components are not clear. This model does not provide specific instructions regarding its implementations. That is why our implementation of the components of the STM is one of the possible implementations[3]. Some of the purely theoretical components are not specified at all, however, a lot of experimental data exists regarding some of them (e.g., working memory).

4. The STM does not explain some of the very important technical issues of the modules. For example, one of the STM's assumptions is that the accumulator collects pulses whenever a stimulus appears. It is not, however, explicitly explained how the accumulator decides that the stimulus ended (particularly, how it is known that the next pulse constitutes the new stimulus). It is also unknown how the information about the amount of pulses should be sent to the working memory. Moreover, it is not explained what happens with information that is already stored in the accumulator (if and when it should be erased, how this mechanism should work). In our implementation, negative feedback loops and other solutions are employed for such purposes, and these solutions can become a matter of debate or research hypotheses.

3.3 Modules in the implementation

For the implementation, we used neuron types that are available in the standard distribution of the Framsticks toolkit [17], with the exception of the Sum neuron. The types of neurons (i.e., available "building blocks" that were used for the implementation of the STM) are shown in Fig. 5. Note that another types of neuronal units may have been used as well; the choice was arbitrary while we tried to take advantage of the most of the built-in, simple neural types, and avoid creating customized, high-level neural units. Obviously, the choice of the building blocks determines the design; using other types of neural units results in a different implementation.

The SeeLight receptor provides information about the magnitude of the stimulus. The Pulse is the generator (it can generate a pulse train according to geometric, normal

[3]Despite the fact that the implementation is only prototypical, the accuracy that has been gained may provide new points for theoretical considerations. The new theoretical and technical solutions presented here can be the basis for new research hypotheses.

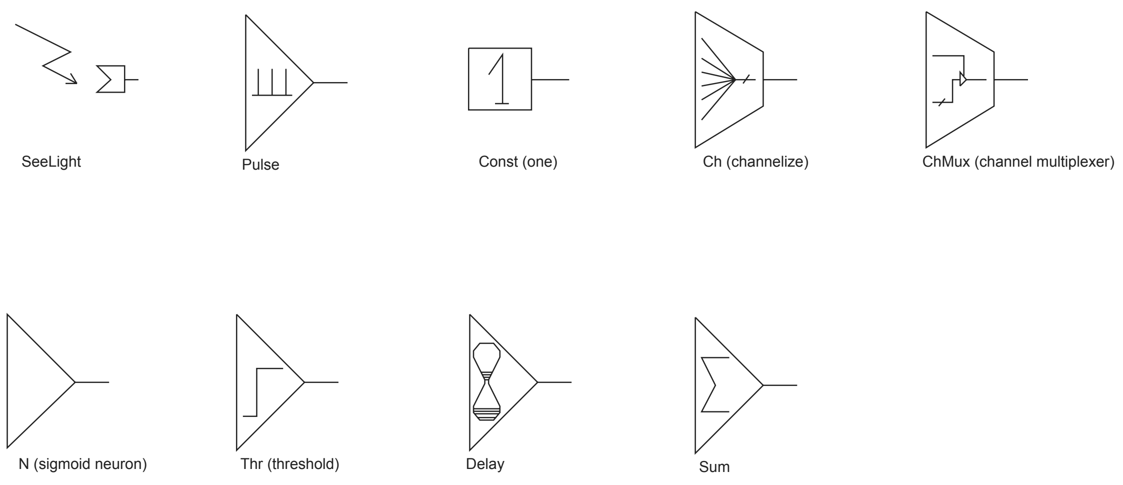

Figure 5: Neuron types used in the implementation of the STM (Fig. 6). See text for details.

or uniform interpulse distributions). The N neuron is a popular AI neuron with a logistic activation function. The Thr neuron outputs either of the two values ("low" or "high") depending on whether the weighted sum of inputs exceeds its threshold value. The Delay neuron delays output to input for a given number of simulation steps. The Sum neuron, in each simulation step, adds a weighted sum of inputs to its internal state, and outputs the state. The Channelize unit packs its input values into a single multichannel output (multiple values are sent in a single connection as shown in Fig. 6), while the ChMux unit uses its control (upper on the graph) input to select one channel from the multichannel (lower on the graph) input.

The numbers of modules presented below correspond to numbers shown in Fig. 6, and it is easier to get the idea of the implementation when one refers the description below to the network diagram. The ordering of modules does not always follow the direction of transmission of information in the network.

1. Pacemaker.

   The pacemaker generates discrete pulses with adjustable delay. The delay can vary according to the distribution specified (uniform with a given range, Gaussian with a given standard deviation, exponential with a given probability).

2. Switch.

   The pacemaker module is connected to the channelize neuron (Ch) marked as letter [A] in Fig 6. There is another connection to the channelize neuron from the pacemaker, but this one with the zero weight, which is equivalent to constantly generating the signal of value zero. This kind of usage of the Ch-neuron is present in a few places in the network. The channelize neuron is connected to the Channel Multiplexer (ChMux) neuron [AX] controlled by the threshold (Thr) neuron [B] connected to the stimulus sensor (in this case, the light receptor: SeeLight). This ChMux-Thr system is considered to be a switch: the appearance of a stimulus causes Thr's output value to be 1. If there is no stimulus, the output value is set

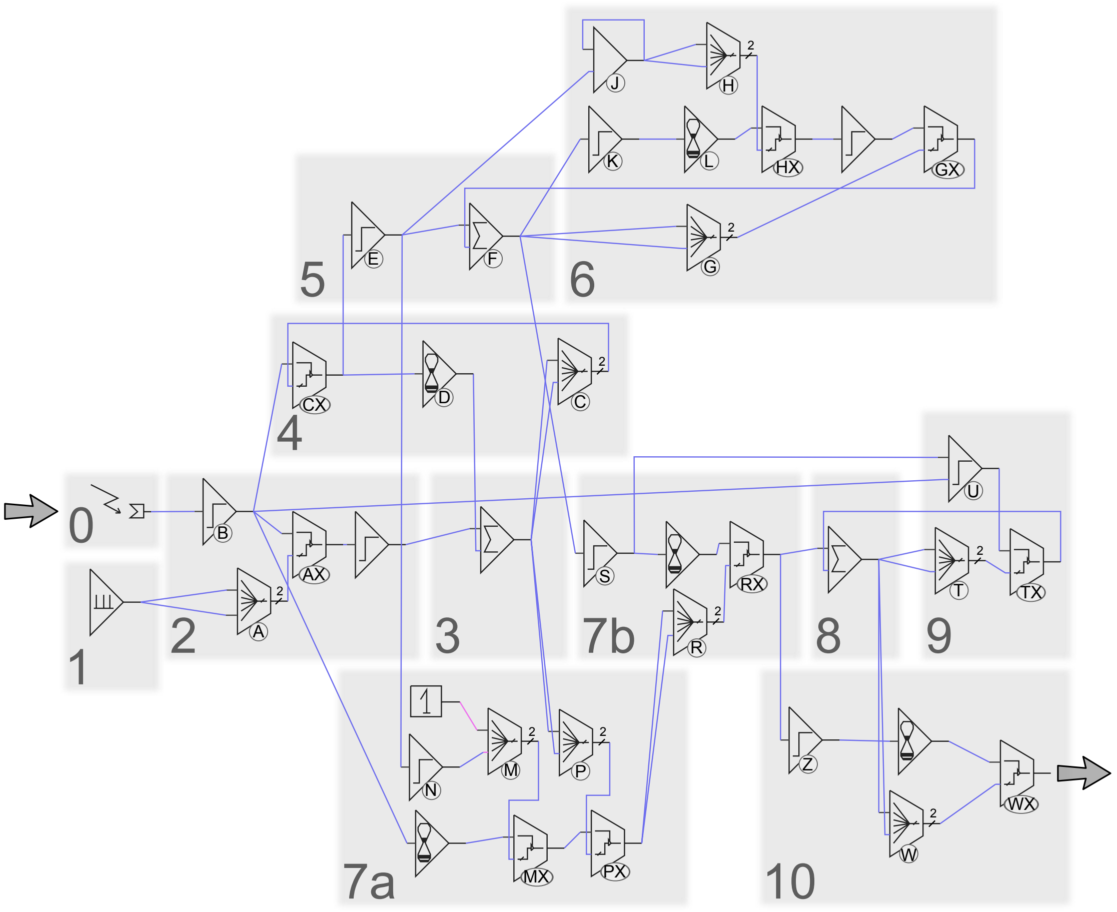

Figure 6: The neural network that implements the Scalar Timing Model. The numbers in this diagram correspond to numbers of paragraphs in Section 3.3. “0” is the stimulus receptor.

to -1. Depending on the output value from Thr, the ChMux passes the signal from the pacemaker module or outputs zero.

3. Accumulator.

   The accumulator is represented by the summing neuron (Sum) which, in each step, adds the input value to its internal state, and sends it to the working memory module. Whether a linear summing neuron or a more complex integrator should be used here is a matter of debate: a sum is the simplest solution, but may have no direct biological equivalent. The contents of the accumulator needs to be cleared somehow in a proper moment, and it should always be prepared for a new stimulus. It is important that the new (second) stimulus does not interfere with the information already stored in the Sum neuron – that could increase the measurement error. To avoid this, a negative feedback loop was implemented that resets the accumulator.

4. The accumulator reset loop.

   The information about the activity of this loop (whether it is working or not) is useful on the higher levels of information processing, primarily in the working memory. The loop is one of the possible approaches to clear the contents of the accumulator. The Sum neuron is connected to the Ch-ChMux subsystem [C] and [CX] with a negative, less than minus one value of weight. There is another connection to this subsystem with the zero weight. The neuron controlling the Ch-ChMux subsystem is the stimulus Thr neuron [B] so that the appearance of a stimulus activates the resetting loop. Between the ChMux neuron [CX] and the module 3 Sum neuron there is a Delay neuron [D] which delays input to output (three simulation steps in this particular case). This neuron makes the resetting process slower, which is required for synchronization with other modules.

5. Process information storage circuit.

   This module does not exist in the theoretical model, but it is a fundamental part of the implementation needed to discriminate between two successively appearing stimuli. It is built from one Thr neuron [E] sending 1 whenever the accumulator resetting loop is working, otherwise sending zero. This Thr neuron is connected to the core of the module, which is the Sum neuron [F] collecting information about the activity of the resetting loop. The Sum neuron is connected to the Accumulator–Working Memory mediator system. This connection is used to set the mode of transmission of the signal from the Accumulator to the Working Memory. The second connection to another part of the mentioned mediator system is used to provide information about when the resetting of the WM should start.

6. Process information storage reset loop.

   As any resetting loop present in the implementation, the process-information storage reset loop is not specified by the STM.

   This reset loop is rather complicated because it has to erase information about a process, not about a stimulus. In general, this loop uses a negative feedback mechanism to erase process information storage a few simulation steps after exposition of the second stimulus, so that another pair of stimuli can be later compared. The negative feedback is provided by the Ch-ChMux system [G] and [GX], to which the Sum neuron [F] is connected with a negative weight of value less than minus one (just like in the case of the accumulator resetting loop). However, controlling the input of this system is complicated and divided into two circuits. These two streams meet in the general controlling unit of the loop – which is another Ch-ChMux subsystem [H] and [HX] connected to the Thr neuron that decides whether to start the erasing process.

   The first stream contains the Thr neuron [E], included in the process information storage. It is gating information about whether accumulator reset loop was active or not. It is important because the process information resetting loop should start working just after the exposition of (the second) stimulus. The accumulator resetting loop provides information about the stimulus exposition. The Thr neuron sends the appropriate signal to the sigmoid neuron [J] which has a low *force* property and zero *inertia* property. These two properties guarantee a low discharge rate of this neuron, so that the provided information persists longer than the operation of the Thr-neuron (which sends signal immediately and does not store it). This information is sent in the next step to the Ch-ChMux subsystem (a general controlling unit – [H] and [HX]).

This subsystem is controlled by the second circuit. It contains the Thr neuron [K] charged by the signal coming from the process information storage Sum neuron [F]. The Thr neuron sends a positive signal only if there is information (about the former work of the accumulator resetting loop) collected in the Sum neuron. The Thr neuron is connected to the Delay neuron [L] with a relatively high delay time, so that the reset loop does not start after exposition of the first stimulus[4]. The Delay neuron controls the Ch-ChMux subsystem [H] and [HX] mentioned at the end of the previous paragraph.

The Thr neuron, gating signal from the general controlling unit [H] and [HX], controls the Ch-ChMux subsystem [G] and [GX]. It should initiate the resetting process when the following two conditions are met:

(a) There is some information in the process information storage.

(b) There is some information in the circuit charged by the accumulator resetting loop.

7. Accumulator–Working Memory Mediator.

   This system is connected with the process information storage. It is created to send accurate information about the amount of pulses stored in the accumulator while the stimulus lasts. Together with the process information circuit, this module can "recognize" whether the first or the second stimulus from a pair is processed. Similarly as the previous module, this one is not mentioned amongst components of the STM and has been created especially for this implementation.

   For simplicity, this module will be described as composed of two submodules.

   (a) The first submodule

       The first neuron of this circuit is the Delay needed for synchronisation with other systems. The Delay neuron is connected to the stimulus Thr neuron [B]. The output signal from Delay controls the Ch-ChMux subsystem [M] and [MX].

       The information input of the Ch neuron [M] comes from another micro-circuit within this module, which consists of the Thr neuron [N] connected to the previous Thr neuron [E] – the one in the storage circuit. The second Thr neuron placed after another threshold is not redundant, because the output value here is needed to be either -1 or $+1$ (in the previous case, 0 or 1 were needed). The Thr neuron [N] in this circuit is connected to the Ch-ChMux subsystem [M] and [MX]. Another neuron plugged into this subsystem is the Constant neuron that provides a stable signal of value one. This neuron is connected to the Ch-ChMux with a negative weight. This micro-system is one of the control systems of the accumulator–working memory mediator which ensures that the information about the total amount of pulses stored in accumulator can be passed further.

[4]The parameters of the Delay neuron and the N neuron decide when the resetting process is activated. The resetting process should start after the second stimulus, but when the accumulator stores a very high value, the resetting might start right after the first stimulus of a pair. This would result in the network reaching its initial state (as if no stimulus was presented), which is a technical drawback and is compensated by a relatively small magnitude of values stored in the accumulator.

The output of the [M] and [MX] Ch-ChMux subsystem controls the input of another Ch-ChMux subsystem [P] and [PX]. The Channelize neuron [P] has an input from the accumulator (Sum) neuron. The Sum neuron has two output connections plugged into Ch – one with weight of value one and another with value zero. These connections are used to provide signal from the accumulator.

(b) The second submodule

This is another branch of the mediator – the one connected with the process information storage circuit.

The output of the Ch-ChMux subsystem from the first submodule ([P] and [PX]) is used as an input to another Ch neuron [R]. There are two outputs from [PX] connected to [R] – one with a negative weight -1, and the other with a positive weight $+1$. The Ch neuron is used as an input to the multiplexer neuron [RX] connected to the Sum neuron, which acts as a Working Memory (to be more precise, it can be seen as a working memory, a reference memory and a comparator as well. Its mode depends on the type of information processed by other components of the neural network).

Another element of this submodule is the Thr neuron [S], to which the Sum neuron [F] from the process information storage is connected. The Thr neuron [S] sends the signal of $+1$ if and only if it has a positive input signal from the Sum neuron [F]. Otherwise, it sends the negative value, -1. The Thr neuron is connected to the Delay neuron which is used to maintain synchronization within the neural network. The Delay is connected as a control input to the Ch-ChMux subsystem [R] and [RX], which is connected to the working memory. In general, the second submodule controls the polarity of the signal coming from the accumulator (through the first submodule). Thanks to this circuit, the network "knows" whether it processes information about the first or the second stimulus from a pair.

8. Working Memory and Reference Memory.

   The core neuron of the memory is the Sum neuron mentioned in the previous section. In the current implementation, it is functionally equivalent to both the working memory and the reference memory. According to the rules implemented in the mediator, when the first signal is presented, the Sum neuron does not collect any information. A few simulation steps after the end of the first stimulus, the Sum neuron gains information stored in the accumulator (here, the Sum neuron acts as a working memory). It maintains this information during the time gap between both stimuli and during the second stimulus – this is a functional equivalent of the reference memory. When the second stimulus ends, the Sum neuron collects information from the accumulator (again acting as a working memory). According to the rules of the mediator, the Sum neuron adds the second signal with a negative sign. As a consequence, the second signal is subtracted[5].

9. Working/Reference Memory reset loop.

[5]There are some reasons to think that a comparison of intervals is not a simple subtraction [11] – psychological effects related to comparison might be by-products of the work of the whole network, not just a single unit. In this implementation the comparison is simplified to subtraction, but this may be changed in the future.

The WM/RM resetting loop mechanism is similar to the accumulator resetting loop. The difference is that the Ch-ChMux subsystem [T] and [TX] is controlled by the Thr neuron [U] that compiles signals from the stimulus Thr neuron [B] and the Thr neuron [S] from the second subcomponent of the accumulator–working memory mediator. The resetting loop Thr neuron [U] activates the resetting process when it gets information that there is no stimulus being presented and that the process information storage is empty (the first stimulus of a pair was not presented). As a by-product of the network structure, shortly after the presentation of the first stimulus, this neuron enters the resetting phase: the resetting loop becomes active, yet it is not needed. This process however does not affect the WM/RM state, because it starts a few simulation steps earlier and stops before the WM collects the value from the accumulator.

10. Comparator.

    The effect of the subtraction process[6], which is the concrete result of subtraction, is then transmitted by the Ch neuron [W] (with two inputs from the working memory Sum neuron, one of them with a zero weight) to the ChMux neuron [WX] controlled by the Thr neuron [Z]. The Thr neuron has an input from the mediatory ChMux neuron [RX], connected to the working memory as well. Between the comparator's ChMux neuron [WX] and the Thr neuron [Z] there is the Delay neuron, needed to maintain synchronization between modules. The output of the [WX] multiplexer provides information about the difference in length of the two stimuli.

The modules described above constitute the implementation of the STM. The information output from the network, i.e. the result of stimulus comparison, may be used further in several ways – for example, to evaluate speed of movement of some objects in the environment.

4 The pacemaker and the accumulator

This section demonstrates the operating principle of one of the most fundamental parts of our STM implementation, which is the pacemaker and the accumulator. The accumulator, implemented as the Sum neuron, adds its input values (coming from the pacemaker's output) to its internal state, and the internal state constitutes the output of this neuron.

Fig. 7 illustrates the way both neurons work. The Pulse neuron generates pulses at specified intervals with the specified distribution and degree of randomness. This particular figure shows the output of the neural network described by the following genotype[7]:

```
X [Pulse, period:20, rand:10] [Sum, -1:1]
```

The first neuron, Pulse, generates a train of pulses of value 1 (this is the default amplitude) with the mean interval of $d = 20$ time steps, and randomness of $\Delta = 10$ time steps (for details, read the following section). Uniform distribution is used here, so the

[6]The subtraction process could also be ascribed to the comparator module.

[7]This is a textual representation of a neural network. One can use it to simulate and test the neural network in the Framsticks simulator [17].

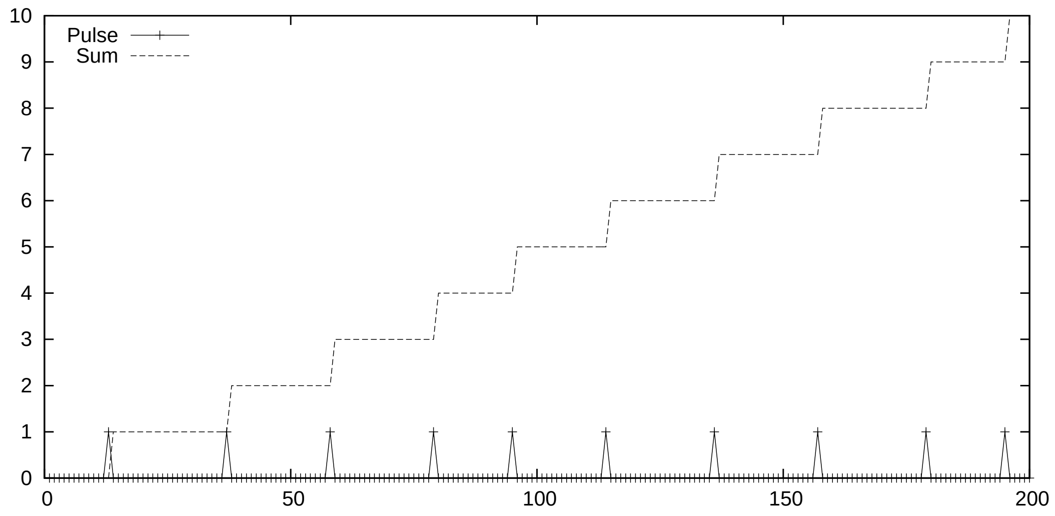

Figure 7: The illustration of a run with the Pulse and Sum neurons. The Sum neuron integrates its inputs in time. Time is shown on the horizontal axis, and output values are shown on the vertical axis.

interval varies from 15 to 25 time steps. The second neuron, Sum, is connected to the previous one (-1 in the genotype) with the connection weight of 1.

The run demonstrates variance of the interval between pulses. In the depicted example, during 200 time steps, the expected number of 10 pulses is generated.

4.1 Properties of the model

This section describes the way the number of pulses n accumulated in the Sum neuron depends on the time of the simulation, t. The experiments follow closely theoretical analyses [14], yet they are performed in an artificial neural network setting where time is discrete.

Consider the two neurons illustrated in Fig. 7. The following simulation lengths t (given in simulation steps) are used: 16, 32, 64, .., 18926, 22712. Each interval t represents the length of a stimulus that is to be estimated based on the number of accumulated pulses; for each t, $x = 100$ runs are performed. We observe the value in the accumulator (Sum) after the simulation is completed, and since the simulation is repeated x times, the average n and variance σ^2 are computed for this value.

4.1.1 Pacemaker as a periodic oscillator

Let's assume that the expected interval between pulses is $d = 20$ time steps, but it varies according to one of the distributions with the standard deviation of σ_d:

- uniform distribution: the length of the interval can be any integer value from $d - \frac{\Delta}{2}$ to $d + \frac{\Delta}{2}$ and all are equally probable; Δ is even and $\sigma_d = \sqrt{\frac{(\Delta+1)^2-1}{12}}$
- normal distribution: in this case, $\sigma_d = \Delta$, so the integer length of the interval follows roughly the distribution of $\mathcal{N}(d, \Delta^2)$; for lengths less than 1, pulses take

place immediately one after another.

In the forthcoming experiments, it has been assumed that $\Delta = \frac{d}{2} = 10$. Figures 8 and 9 show the average, variance σ^2, and the variance-to-average ratio of the accumulated values, depending on the simulation time t (horizontal axis). The average value in the accumulator corresponds to the expected number of pulses n in the given time period t. The variance of the accumulated value grows linearly with respect to the simulation time so that their ratio remains constant, with the exception of short periods where the relative variance is much higher.

The general shape of characteristics presented in Figures 8 and 9 does not depend on the pulse interval length or the amount of its randomness. In the two experiments, for long time periods, the variance ratio σ^2/n of the accumulated value n for the normal distribution of pulse intervals oscillates around 23–25%, which is approximately 10 times higher than for the uniform distribution. This is consistent with earlier analytical results [14] that expect $\lim_{n\to\infty} \frac{\sigma^2}{n} = c$. The $c = \left(\frac{\sigma_d}{d}\right)^2$ parameter characterizes the oscillator; for normal distribution used here, ${\sigma_d}^2 = 100$ and $c = 100/400 = 0.25$. For uniform interpulse distribution, ${\sigma_d}^2 = 10$ and $c = 10/400 = 0.025$.

4.1.2 Pacemaker pulse generation as the Poissonian process

The pacemaker would also serve its purpose if it was based on events occurring randomly, with a constant probability. This would also yield a constant mean number of pulses in a time frame, yet their distribution in time would not be as ordered as with the two types of periodic oscillators considered above. Since the neural network simulation operates in discrete time, generating pulses would be in fact a Bernoulli process. Intuitively, such a pacemaker would be less suitable for measuring time than earlier periodic pacemakers because of the higher variance.

Figure 10 presents the experiment performed with the Bernoulli pacemaker: in each simulation step, the probability of generating a pulse is $p = 1/20$ which corresponds to $d = 20$ in experiments from Sect. 4.1.1. Just like in the two previously analyzed examples, here the average follows closely the expected mean number of events (pulses). This is a desired property, since it allows to measure time with no bias. The variance of n is close to n, so the σ^2/n ratio varies around 1; in theory [14] we expect it to be $1 - p = 0.95$. Therefore, this pacemaker is less precise than the periodic ones.

4.2 Discussion

The results from Sect. 4 show that the coefficient of variation (that is the standard deviation to mean ratio) of values stored in the accumulator decreases rapidly for short durations. This is inconsistent with the formal scalar property (the coefficient of variation should remain constant), but it is corroborated by empirical data [23]. Just as it has been expressed by Wearden and Lejeune [27]:

> Almost all studies that have used durations less than 100 ms have reported increased relative variance (i.e., increased CV or Weber fraction-like measure) at very short durations compared with longer durations.

Such a violation of the Weber's law is also consistent with the data collected across many experiments studying perception mechanisms of different modalities. This vi-

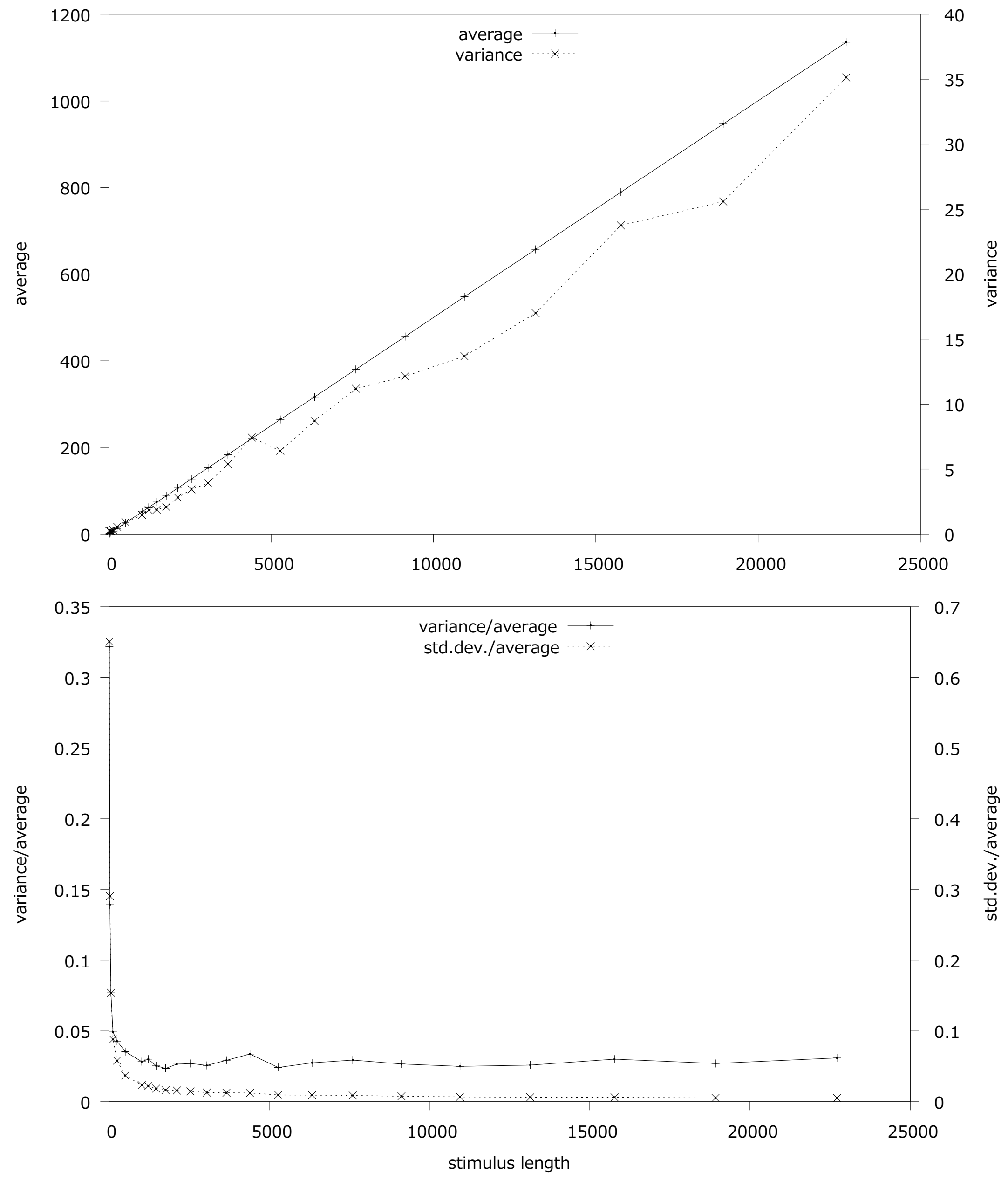

Figure 8: Uniform distribution of randomness of the pulse interval. Accumulated value (average, variance, standard deviation, and their ratios) versus simulation time (stimulus length), t.

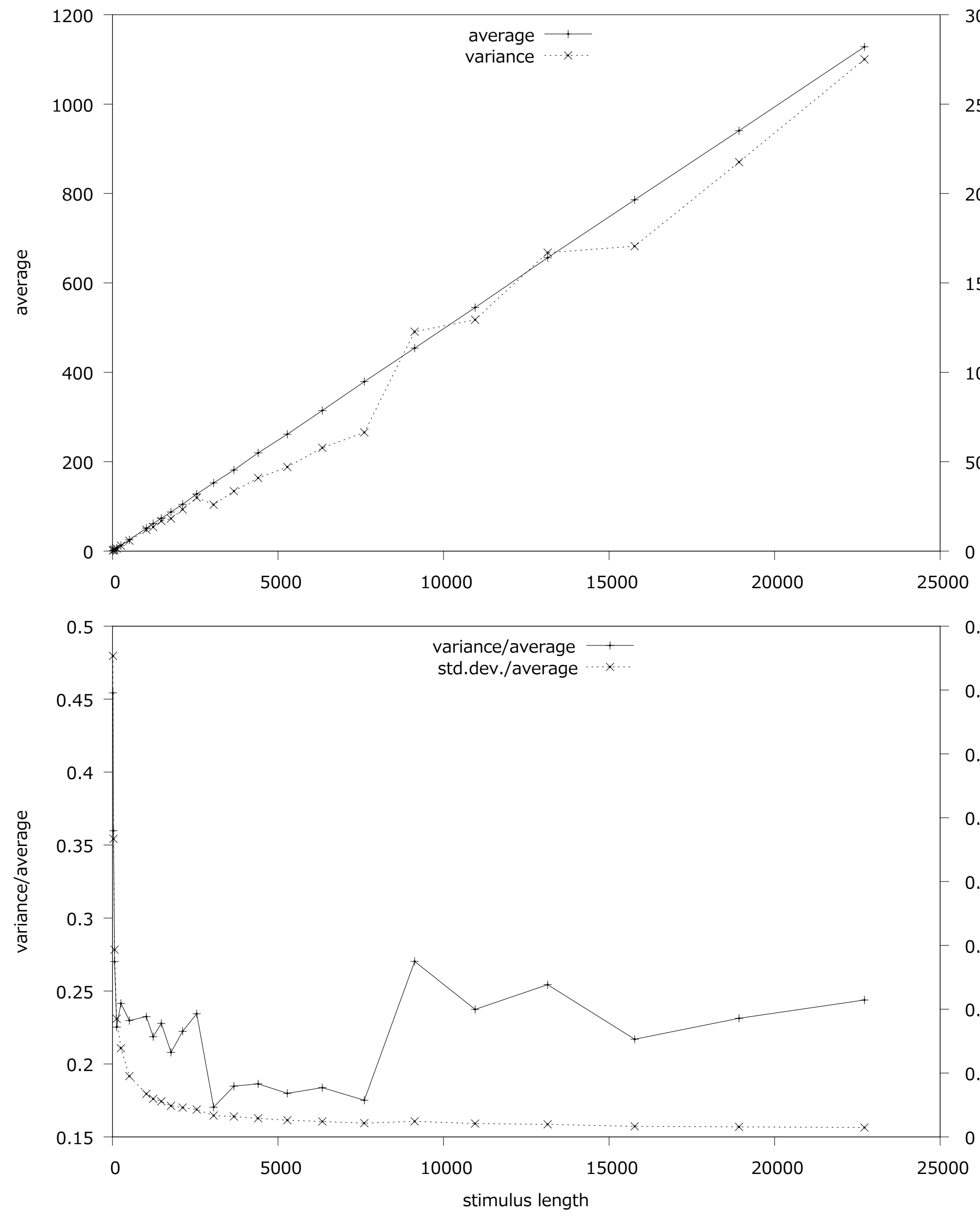

Figure 9: Normal distribution of randomness of the pulse interval. Accumulated value (average, variance, standard deviation, and their ratios) versus simulation time (stimulus length), t.

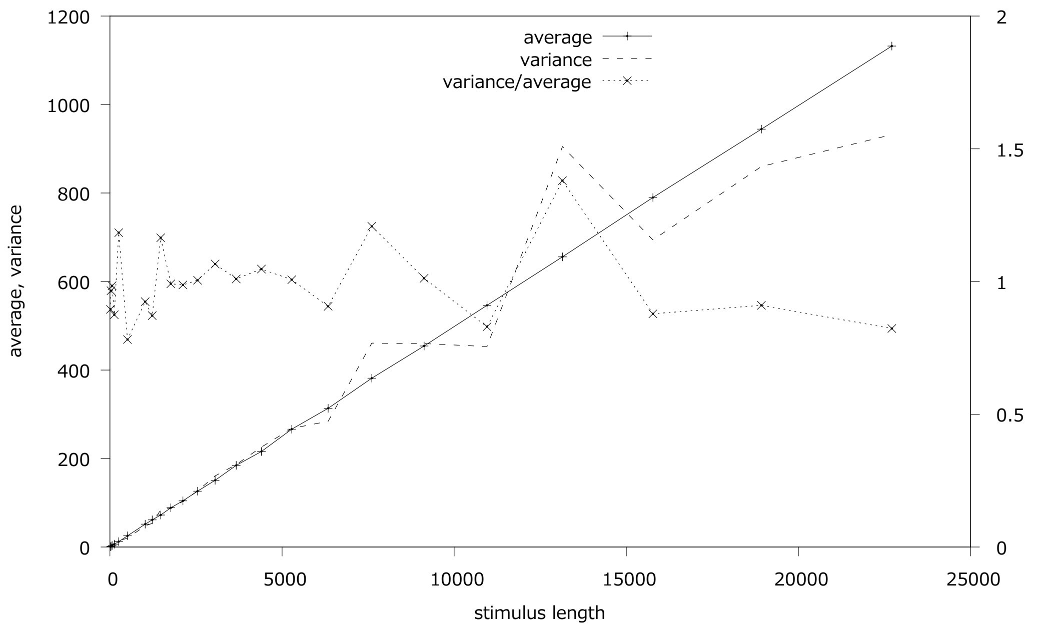

Figure 10: A discrete model of a Poissonian pacemaker. Accumulated value (average and variance) versus simulation time (stimulus length), t.

olation led to a change in the formalized Weber's law formula and is interpreted by psychophysicists as an induced sensory noise [3].

Apart from the reasons behind this phenomenon, the phenomenon itself is a bit problematic to the Scalar Timing Model. The work [27] suggests a possible explanation that saves the scalar properties of the STM and interprets the effect of rapid increase of the coefficient of variation for short intervals as the delay of the opening/closing of the switch related to the stimulus appearance – which is a non-scalar source of variance. This explanation needs to be more precise to be able to predict the results of timing in humans (e.g., what determines the latencies and what is the scale of this effect). The results of experiments described here are consistent with the properties of the clock-counter models: the reason for the rapid drop of the coefficient of variation for short durations is the pacemaker mechanism itself.

The experiments performed in this work show that the coefficient of variation of values stored in the accumulator drops slowly for longer durations. This is inconsistent with the formal assumption of the scalar property of the model, and with some part of experimental data collected across experiments, when the coefficient of variation or the Weber fraction are taken into consideration [2, 10]. In other experiments however, the coefficient of variation does not remain stable and drops as a function of stimulus duration [30, 18], just as in our simulations. Also in [25] (pages 161–162) such drops are reported, yet only in a limited range of base durations. This short analysis of existing results does not suggest a specific explanation for this drop, only a general one, such as a sensory-dependent bias. In another related work, Killeen and Weiss (especially pages 462–463 of [13]) discussed the mechanism of the counting clock-timer. They also referred to the experimental data of brief intervals to show that a very general description may be applied to results regarding different time scales.

The interpulse distributions used in the experiments in this work (exponential, normal, uniform) model the distributions that can occur in real oscillators. The scalar property demands that the coefficient of variation is constant independently of the duration of the stimulus, yet this is not always observed in experimental data: within some ranges of stimulus duration it is hard to recognize the coefficient of variation decreasing as a function of stimulus duration. There may be not enough data collected across individual experiments, and differences may be too small (smaller than the measurement error within this range of stimuli), therefore statistical tests do not confirm that there is a decrease of the coefficient of variation as a function of the time of stimulus exposition. Attempts to study time perception for a wide range of durations [18] demonstrated that the coefficient of variation decreases with the increase of duration of examined intervals, which is consistent with our results. The work [18] suggests that in typical experiments regarding a small range of intervals, the differences between sensitivities related to timed intervals are too small to notice a decrease. This means that in some experiments, there are likely reasons to observe a locally constant coefficient of variation even though it decreases [14]. Moreover, due to the limited number and randomness in experiments, in selected ranges of stimulus durations the data may suggest an increase of the coefficient of variation, or periodic changes.

To perform direct comparisons with real-life experimental data, the time scale of the simulation needs to be adjusted to the real time scale. This would allow for more meaningful predictions and would facilitate integration of results from various biological and psychological experiments – in particular, it would allow to refer the probability distributions of the pacemaker to the existing experimental data. Obtaining and integrating psychophysical and neurobiological data into the implementation (e.g., acquiring knowledge about the mechanisms of oscillatory neurons) will increase utility of the implementation.

For the simulations presented in this section, the variance-to-mean ratio was an important indicator of timing sensitivity. Whichever of the distributions of pulse intervals is used, for long durations it captures a specific property of timing [4]: as presented in figures 8, 9 and 10, the ratio remains constant in a wide range of stimulus durations, with an exception of short durations for periodic pacemakers [14]. For further discussions on this topic and the implementation of the scalar property, see [15].

5 The STM implementation in action

For the implementation shown in Fig. 6, a sample run and outputs of the key neurons are shown in Fig. 11. The pattern presented in this sample run represents the three major processes of the STM implementation. The accumulation of the pulses by the accumulator is represented by the solid line in the top panel. Note that after the end of each stimulus (stimuli are shown as the dashed line), the accumulator is reset in order to be able to store information about another stimulus. The working memory is represented by the solid line in the middle panel. The value stored in the working memory after the exposition of the first stimulus is equal to[8] the amount collected by the accumulator.

The information about the duration of the first stimulus is stored during the ISI

[8]As mentioned before, the working memory does not always have to directly reflect the accumulator value. For example, it might be multiplied by a random variable [5].

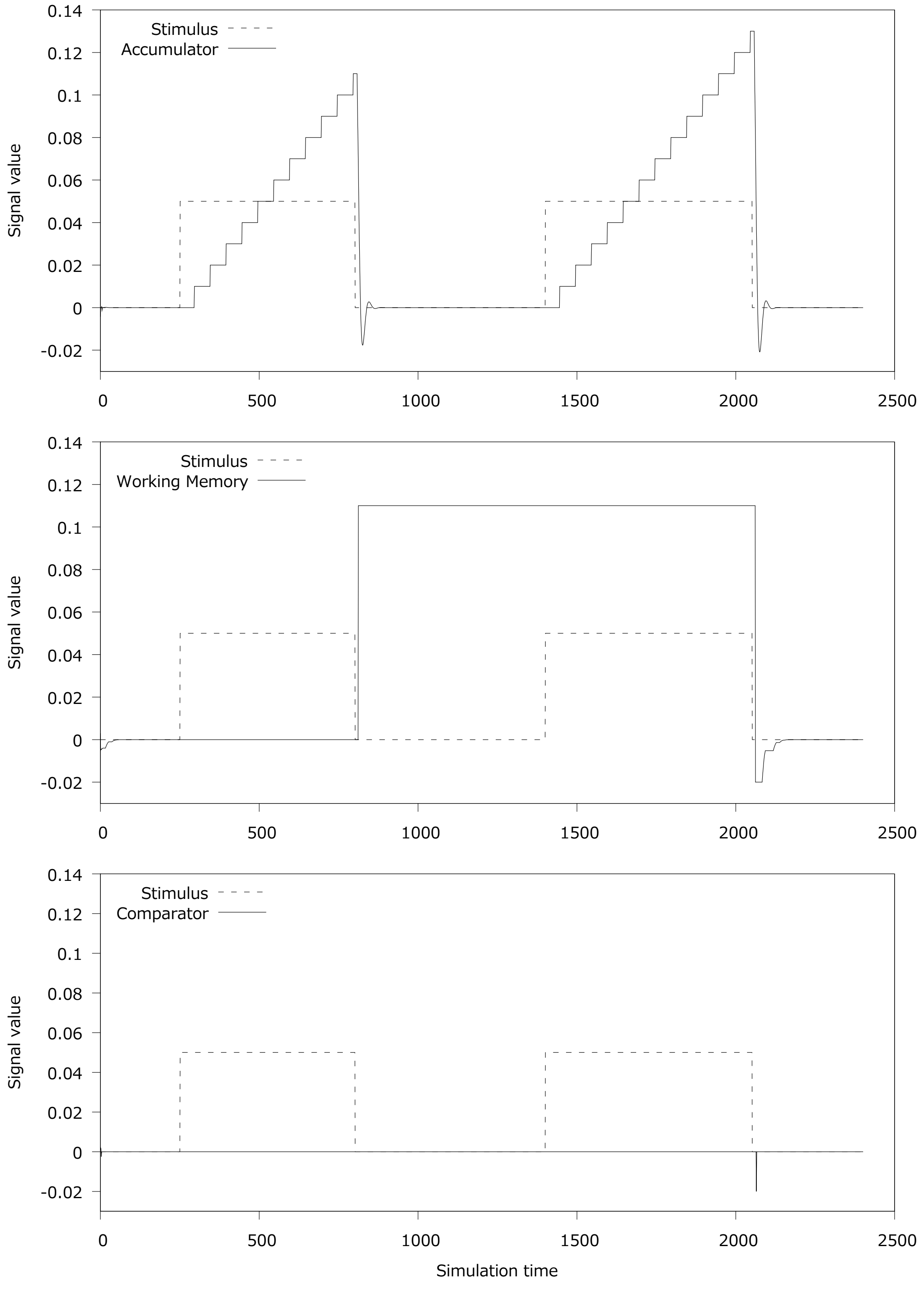

Figure 11: The STM implementation at work.

(interstimulus interval). After the end of the exposition of the second stimulus, the value of the signal is subtracted in the working memory, as seen in the middle panel. Since the second stimulus was longer, the value in the working memory drops below zero. Shortly after the end of the second stimulus, the working memory is reset (the signal sent by the working memory is zero). The bottom panel demonstrates the comparator output (solid line) – the difference between the length of the two stimuli is emitted shortly after the second stimulus ends.

5.1 The STM embodied and situated

The proposed implementation can be used in a number of settings and circumstances (e.g., in an artificial creature or a robot following stimuli that last longer or are more stable in the environment). The influence of the length of stimuli on behaviors depends on particular goals and applications, and can be easily implemented in the simulation where both brains and bodies are considered.

6 Conclusions

The results regarding the pacemaker-accumulator module demonstrate that independently from the interpulse distribution, the variance-to-mean ratio drops quickly for short stimulus durations and remains constant for long durations. Additional mechanisms (variable pacemaker pulse rate, specific characteristics of the switch, or others) may be introduced in this implementation to obtain the scalar property [5, 4, 6]. The simulation experiments with the current implementation demonstrate violations of the scalar property [27], and illustrate potential difficulties in inferring relationships describing behavior of the model based on imperfect and scarce data [18, 14].

The flexibility of artificial neural networks in our implementation, along with available information about psychology, psychophysics, and physiology of time, creates great opportunities to verify hypotheses regarding individual components (or even individual circuits within the components), as well as hypotheses about the relations of groups of components. It is also possible to study the implemented STM as a whole. This can be useful to find out which processes are responsible for some of the observed psychological effects (e.g., the time-order error [11]).

What is more, a sample and prototype implementation presented here creates opportunities to generate original research hypotheses inspired by the structure and design of the artificial neural network. Even if some hypothesis inspired by the implementation is not valid, the flexibility of structure of the implemented STM allows for easy modifications according to the new, acquired information (just as it is possible to adjust the distribution of pulses to fit experimental data). Such modifications may result from new facts that are likely to appear in experimental sciences.

In summary, this paper describes one of the possible implementations of the Scalar Timing Model. Independently from this particular implementation, research has been conducted to find out how is the distribution of pacemaker pulses related to stored information about the duration of the stimulus. Three interpulse probability distributions were selected in an attempt to meet theoretical demands and follow experimentally collected data. The next step is to integrate more information about the timing process, and find possible explanations of the observed effects. The potential conclusions from these experiments may influence the original definition of the STM. Ongoing experiments include using evolutionary algorithms for optimization, and the artificial world environment for embodiment of the working STM. This implementation, after successful development, may become a frame of exchange and integration of information on the verge of many scientific domains.

References

[1] Russell M. Church. Evaluation of quantitative theories of timing. *Journal of the Experimental Analysis of Behavior*, 71(2):253–256, 1999.

[2] Russell M. Church. A concise introduction to scalar timing theory. In Warren H. Meck, editor, *Functional and Neural Mechanisms of Interval Timing*, pages 3–22. CRC Press, Boca Raton, Florida, 2003.

[3] George A. Gescheider. *Psychophysics: The Fundamentals*. Lawrence Erlbaum Associates Inc, 3 edition, 1997.

[4] David J. Getty. Discrimination of short temporal intervals: A comparison of two models. *Perception & Psychophysics*, 18(1):1–8, 1975.

[5] John Gibbon. Ubiquity of scalar timing with Poisson clock. *Journal of Mathematical Psychology*, 35:283–293, 1992.

[6] John Gibbon. Multiple time scales is well named. *Journal of the Experimental Analysis of Behavior*, 71:272–275, 1999.

[7] Simon Grondin. From physical time to the first and second moments of psychological time. *Psychological Bulletin*, 127(1):22–44, 2001.

[8] Simon Grondin. Overloading temporal memory. *Journal of Experimental Psychology: Human Perception and Performance*, 31(5):869–879, 2005.

[9] Simon Grondin. Methods for studying psychological time. In Simon Grondin, editor, *Psychology of time*, pages 51–74. 2008.

[10] Simon Grondin, Bastien Ouellet, and Marie-Éve Roussel. About optimal timing and stability of Weber fraction for duration discrimination. *Acoustical science and technology*, 22(5):370–372, 2001.

[11] Åke Hellström. Comparison is not just subtraction: Effects of time- and space-order on subjective stimulus difference. *Perception & Psychophysics*, 65(7):1161–1177, 2003.

[12] Christian Kaernbach. Simple adaptive testing with the weighted up-down method. *Perception & Psychophysics*, 49:227–229, 1991.

[13] Peter R. Killeen and Neil A. Weiss. Optimal timing and the Weber function. *Psychological Review*, 94(4):455–468, 1987.

[14] Maciej Komosinski. Measuring quantities using oscillators and pulse generators. *Theory in Biosciences*, 131(2):103–116, 2012. URL: `http://dx.doi.org/10.1007/s12064-012-0153-4`, `doi:10.1007/s12064-012-0153-4`.

[15] Maciej Komosinski and Adam Kups. Time-order error and scalar variance in a computational model of human timing: simulations and predictions. *Computational Cognitive Science*, 1(3):1–24, 2015. URL: `http://dx.doi.org/10.1186/s40469-015-0002-0`, `doi:10.1186/s40469-015-0002-0`.

[16] Maciej Komosinski and Szymon Ulatowski. Framsticks: Creating and understanding complexity of life. In Maciej Komosinski and Andrew Adamatzky, editors, *Artificial Life Models in Software*, chapter 5, pages 107–148. Springer, London, 2nd edition, 2009. URL: `http://www.springer.com/978-1-84882-284-9`.

[17] Maciej Komosinski and Szymon Ulatowski. Framsticks website, 2021. `http://www.framsticks.com`.

[18] P.A. Lewis and R. C. Miall. The precision of temporal judgement: milliseconds, many minutes and beyond. *Philosophical Transactions of the Royal Society B*, 364(2):1897–1905, 2009.

[19] Pete Mandik. Varieties of representation in evolved and embodied neural networks. *Biology and Philosophy*, 18(1):95–130, 2003. URL: `http://testweb.wpunj.edu/cohss/philosophy/FACULTY/mandik/papers/vreenn.pdf`.

[20] Pete Mandik, Mike Collins, and Alex Vereschagin. Evolving artificial minds and brains. In Andrea C. Schalley and Drew Khlentzos, editors, *Mental States – Volume 1: Evolution, function, nature (Studies in Language Companion, Series 92)*. John Benjamins Publishing Company, 2007.

[21] Matthew S. Matell and Warren H. Meck. Cortico-striatal circuits and interval timing: coincidence detection of oscillatory processes. *Cognitive Brain Research*, 21:139–170, 2004.

[22] Paolo Nichelli. The processing of temporal information in the frontal lobe. In Francois Boller and Jordan Grafman, editors, *Handbook of Neuropsychology: The Frontal Lobes*, chapter 9. Elsevier, 2002.

[23] Thomas Rammsayer and Rolf Ulrich. Counting models of temporal discrimination. *Psychonomic Bulletin & Review*, 8(2):270–277, 2001.

[24] Thomas H. Rammsayer. Ageing and temporal processing of durations within the psychological present. *European Journal of Cognitive Psychology*, 13(4):549–565, 2001.

[25] Thomas H. Rammsayer and Simon Grondin. Psychophysics of human timing. In Robert Miller, editor, *Time and the brain*, pages 157–168. Harwood Academic Publishers, 2000.

[26] J. E. R. Staddon and J. J. Higga. Time and memory: Towards a pacemaker-free theory of interval timing. *Journal of Experimental Psychology: Animal Behavior Processes*, 71(2):215–251, 1999.

[27] J. H. Wearden and Helga Lejeune. Scalar properties in human timing: Conformity and violations. *The Quarterly Journal of Experimental Psychology*, 61(4):569–587, 2008.

[28] John H. Wearden. Applying the scalar timing model to human time psychology: Progress and challenges. In Hede Helfrich, editor, *Time and mind II: Information processing perspectives*, pages 21–39. Hogrefe & Huber Publishers, Cambridge, Massachusetts, 2003.

[29] John H. Wearden and M. F. Doherty. Exploring and developing a connectionist model of animal timing: Peak procedure and fixed-interval simulations. *Journal of Experimental Psychology: Animal Behavior Processes*, 21(2):99–115, 1995.

[30] John H. Wearden, Roger Norton, Simon Martin, and Oliver Montford-Bebb. Internal clock processes and the filled-duration illusion. *Journal of Experimental Psychology: Human Perception and Performance*, 33(3):716–729, 2007.

[31] Dan Zakay, Richard A. Block, and Yehoshua Tsal. Prospective duration estimation and performance. In Daniel Gopher and Asher Koriat, editors, *Attention and Performance XVII*, pages 557–580. MIT Press, Cambridge, Massachusetts, 1999.