

Contents

1	Introduction	2
2	State Machine	3
2.1	Inter-Trial Interval	3
2.2	Start Trial	4
2.3	Fixation Time	4
2.4	Stimulus	4
2.5	Choice	4
2.6	Reward	4
2.7	Abort	4
3	Bonsai Guidelines	5
3.1	State (SelectMany) Workflow Organization Logic	5
3.2	Timestamping Method	5
4	Input/Output Files	7
4.1	Animal-Specific File	7
4.2	Task/Level-Specific File	10
4.3	Output File	13

1

Introduction

This is the documentation for the Bonsai workflow developed for the Sound Lateralization Task that is going to be performed by the Circuit Dynamics and Computation group at the Champalimaud Foundation. The goal of this document is to complement the information present in the README of the GitHub repository of the task.

The idea is that a user can easily download and setup the task just by reading the README, but that the implementation details of the task are also documented somewhere in case there is a need to understand them.

The experimental setup makes use of the capabilities of the Harp devices (which implement the Harp protocol) and the Bonsai visual programming language, which work really well together.

A custom Bonsai package called CDC.SLTUtils was developed. This package adds new nodes with functionality which was difficult to implement or wasn't available natively in Bonsai. The source code is also available in the task's GitHub repository in the *./package* folder.

2

State Machine

The task that is going to be described throughout this document was designed as a state machine, where the progression through the different states is driven by certain events. Figure 2.1 shows the state machine that describes this task.

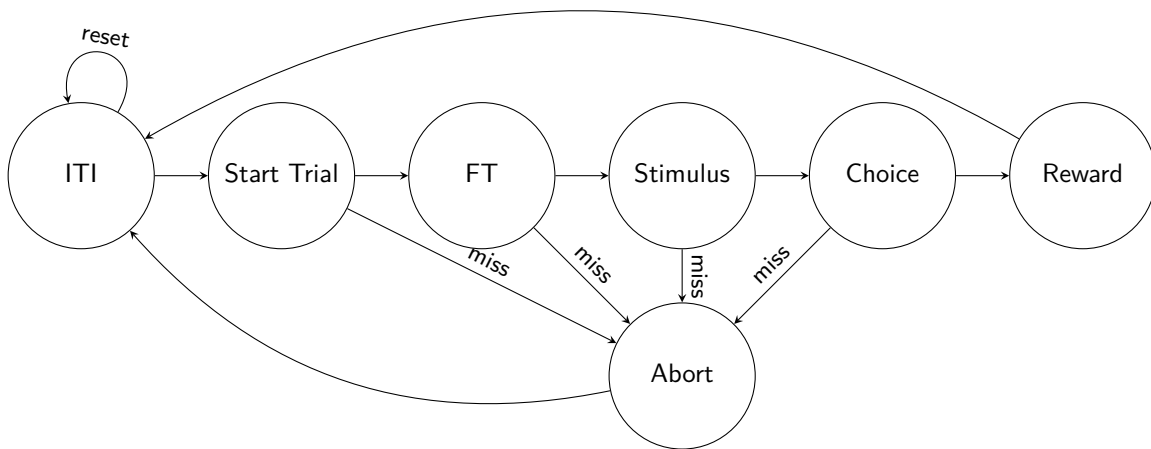


Figure 2.1: State machine of the task

From Figure 2.1, it is possible to see that from some states (Start Trial, FT, Stimulus, Choice) there are two states that these states can progress to. This happens because there was a need to implement a way to abort the current trial of the task in case a certain condition is not met in each of these states. The rest of the chapter consists of a brief description of what happens in each state, in which the explanation of the progression conditions is included.

2.1 Inter-Trial Interval

The Inter-Trial Interval (ITI) state is, as the name suggests, the time interval that separates two consecutive trials. Since there is a need to setup each trial (for example, to (re)set some parameters) and the duration of the ITI is, typically, a few seconds, this state is also used to prepare the new trial. Currently, the actions that take place in the ITI state are:

- Checking if a block of trials ended in the previous trial and, if so, reset the block variables and update the block number and training level;
- Updating the trial number;
- Randomizing the ABL (average binaural level) and ILD (inter-aural level) values and selecting the sound that is going to be played in the current trial.

2.2 Start Trial

A trial starts when the rodent pokes his nose in the central nose port (CNP). So this state consists of waiting that the rodent starts poking the CNP. If there is a poke within a certain time limit (for example, 6 seconds), the task continues as it is supposed to, otherwise this trial is aborted.

2.3 Fixation Time

This is the state that precedes the stimulus presentation. The rodent should stay in the CNP during the entire time this state lasts so that the task progresses as expected, otherwise the trial is aborted. The fixation time varies from trial to trial - it is randomly generated - in order to make the timing of the stimulus presentation (which happens as soon as the fixation time ends) unpredictable. If the stimulus presentation timing was predictable, the resultant reaction times would be affected (and biased). The fixation time is given by:

$$t_{\text{Fix}} = t_{\text{Base Fix}} + (X \sim \text{Exp}(\lambda)) \quad (2.1)$$

2.4 Stimulus

This is the state where the stimulus is presented. The stimulus stops when either the rodent leaves the CNP or when the defined presentation time elapses (the presentation time is usually of a few seconds). It is possible to define a minimum reaction time. If the timed reaction time is less than the minimum required, the trial is aborted.

2.5 Choice

When the stimulus presentation stops, the animal has to pick the side (left or right) which corresponds to the speaker which played the sound the loudest by poking in either lateral nose port. If the animal doesn't decide within a certain time (typically, of a few seconds), the trial is aborted.

2.6 Reward

The Reward state evaluated whether the rodent got the answer right or not. In case the answer is wrong, a penalty time is applied (10 seconds for instance). If the answer is right, the animal only gets the reward (water) if he stayed in the correct LNP for at least a minimum amount of time.

2.7 Abort

As mentioned before, the Abort state is triggered when some conditions are not met in the previous states. This state consists of a small time penalty (typically about 1 second).

Since this state is one of the two possible final states of a trial, there is a need to set/update some variables that would normally be set/updated in states that the state machine did not get into during the current trial.

3

Bonsai Guidelines

The implementation of the task described in Chapter 2 was made with the Bonsai visual programming language. The goal of this chapter is to approach some aspects regarding the Bonsai implementation. However, a minimum degree of familiarity with the language will be assumed and the workflow will not be documented node by node.

When the workflow is first opened, the state machine schematized in Figure 2.1 is easily identifiable. Here are some notes regarding the main workflow:

- The states are implemented as SelectMany nodes;
- There is an additional SelectMany node that is responsible for outputting data from each trial;
- The last nodes of the state machine implementation are what allow the repetition of the workflow and, hence, to start a new trial;
- The initialization of variables, distributions and interface with hardware is implemented in GroupWorkflow nodes that aren't part of the state machine.

3.1 State (SelectMany) Workflow Organization Logic

It doesn't take much until a Bonsai workflow starts getting complex and, consequently, confusing. In order to improve the readability of the workflow (or at least try), a few guidelines are being followed so that any person that needs to look at the implementation of the task can easily understand it (and/or modify it):

- In the main workflow, the inputs and outputs of each SelectMany node (i.e. each state) should be of the type `Tuple<Boolean,int>`. The idea is that the Boolean gives information regarding the validity of a trial, that is if a True comes out of a state the trial should proceed as planned, otherwise it should abort; and the int indicates the state from where the Tuple came from. Despite not being possible to abort a trial from every state, this guideline is a way to future-proof and standardize data-transfer between states.
- Separate independent functionality in a workflow (whether it is the main workflow or one of the SelectMany nodes) should be grouped in GroupWorkflow nodes and displayed in the first line of the workflow.
- If some logic is composed of multiple nodes and is used more than once, it should also be grouped in a GroupWorkflow node to avoid repetitions (example: TimestampEvent node).
- Since there is a sense of sequentiality of events throughout the task, different branches of a workflow should be sorted from top to bottom by order of execution when possible.

3.2 Timestamping Method

In reaction time experiments, it is of extreme importance to timestamp every event with precision. A reliable way to timestamp each event is by using hardware timestamps (from the Harp Behavior board, to be exact).

Inside the SelectMany nodes where the different states are implemented, it is possible to find a node called TimestampEvent after events that need to be timestamped externally (i.e. non-Harp events). This node consists of a GroupWorkflow

which, internally, receives an event from the outside and a hardware timestamp from the Timestamp subject (initialized in the Behavior GroupWorkflow), then “zips” both and sends the tuple to the CreateTimestamped node, which converts the tuple into a `Bonsai.Harp.Timestamped<T>` data type, and finally outputs it to the next node of the workflow. Figure 3.1 shows the inside of the TimestampEvent node.

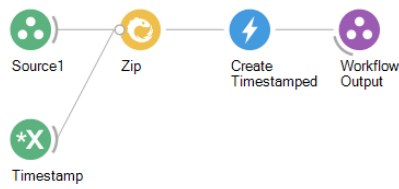


Figure 3.1: TimestampEvent GroupWorkflow (DEPRECATED)

4

Input/Output Files

One of the goals behind this Bonsai workflow is that it is possible to perform similar SLT tasks with small subtleties between them. To achieve this goal, the workflow reads 2 input files (**3 in the future??**) which contain the parameters needed to run the task. The input parameters are divided in 2 (**3 in the future??**) different files: *animal_settings.json* and *training_settings.csv* (**possible setup_settings.json in the future**). By separating the input parameters in different files, it is possible to define the animal-specific(, **the setup-specific**) and the task/level-specific parameters independently.

This chapter is dedicated to the description of the file formats as well as of their parameters. In addition to the input files mentioned previously, the format of the output file (which is only one and universal for all tasks ran with this workflow) will also be described here.

4.1 Animal-Specific File

The animal-specific file is a JSON file which contains information regarding the task that is specific to a certain animal, such as the animal ID number and the session number. Below, it is possible to see an example of this file.

```
{
  "Animal": 1,
  "Box": 2,
  "Session": 53,
  "SessionType": 1,
  "PossibleABLS": [20, 40, 60],
  "CycleILD": 1,
  "Bias": 0.5,
  "FixationBase": 0.01,
  "FixationBaseDelta": 1E-3,
  "FixationBaseTarget": 0.2,
  "FixationExpMean": 0.075,
  "FixationExpMeanDelta": 0.075,
  "FixationExpMeanTarget": 0.075,
  "MinRT": 0.01,
  "RTDelta": 1E-3,
  "RTTarget": 0.15,
  "MaxRT": 10,
  "MaxSamplingTime": [],
  "MinMT": 0.01,
  "MinLNP": 0.01,
  "LNPDelta": 1E-3,
  "LNPTarget": 0.01
}
```

The names and order of appearance of the parameters need to be exactly as shown in the example so that Bonsai is correctly able to load the parameters. For the rest of this section, a description of each parameter contained in the animal-specific file will be made.

Animal

- Description:
- Implemented: Yes

Box

- Description:
- Implemented: Yes

Session

- Description:
- Implemented: Yes

SessionType

- Description:
- Implemented: Yes

SessionDuration

- Description: The duration of the task ("hh:mm:ss").
- Implemented: Yes

StartingTrialNumber

- Description:
- Implemented: Yes

StartingBlockNumber

- Description:
- Implemented: Yes

ABLList

- Description: Possible ABLs to present when DifferentABLs is 1.
- Implemented: Yes

CycleILD

- Description: Random draw from subcycle only to compute next ILD.
- Implemented: Yes

Bias

- Description: Left < 0.5, Right > 0.5.
- Implemented: No

MinFT

- Description: The minimum fixation time (ms).
- Implemented: Yes

FTDelta

- Description: The increment to make to the constant part of fixation time every non-abort trial (ms).
- Implemented: Yes

FTTarget

- Description: The target value for the constant part of the fixation time (ms).
- Implemented: Yes

ExpFTMean

- Description: The mean value of the random part of the fixation time (ms).
- Implemented: Yes

MinRT

- Description: Minimum amount of time in CNP to wait after the sound presentation starts (s).
- Implemented: Yes

RTDelta

- Description: The increment to make to RT every non-abort trial (s).
- Implemented: Yes

RTTarget

- Description: The target value for RT (s).
- Implemented: Yes

MaxRT

- Description: Maximum amount of time in CNP to wait after the sound presentation starts (s).
- Implemented: Yes

MaxSamplingTime

- Description:
- Implemented: No

MinMT

- Description: Minimum time allowed to move to LNP after leaving CNP (s).
- Implemented: Yes

MinLNP

- Description: Minimum poke duration in LNP (s).
- Implemented: Yes

LNPDelta

- Description: The increment to make to LNP time every non-abort trial (s).
- Implemented: Yes

LNPTarget

- Description: The target for LNP time (s).
- Implemented: Yes

4.2 Task/Level-Specific File

The task/level-specific file is a CSV file which contains the input parameters which are specific to a certain experimental procedure but not dependent on the animal. The choice of using a CSV file to input the task/level-specific parameters is justified by the need to easily define different levels (with different parameter values), since an animal has to be trained to perform a specific task, which is achieved by just adding a new row to the CSV file. The last row of the file is the level in which the animal is considered to be fully trained.

For the rest of this section, a description of each parameter contained in the task/level-specific file will be made for order of appearance. Again, the names and order of appearance of the parameters should be preserved to correctly load the parameters to the workflow.

Level

- Description:
- Implemented: No

TrialsPerBlock

- Description: Number of trials of the current block.
- Implemented: Yes

FixedABL

- Description: ABL value to use when DifferentABLs is 0.
- Implemented: Yes

DifferentABLs

- Description: Whether to use the ABLs from the ABLList (1) or the FixedABL (0).
- Implemented: Yes

ABLBlock

- Description: Whether ABLs change only across blocks (1) or not (0).
- Implemented: Yes

ILDStepSize

- Description: The separation between two consecutive |ILD| values.
- Implemented: Yes

ILDSteps

- Description: The number of |ILD| values. The final array will contain 2*ILDSteps elements to account for both the positive and the negative ILD values.
- Implemented: Yes

UseLog

- Description: Whether to use logarithmic steps between consecutive ILD values.
- Implemented: Yes

LogBase

- Description: The base of the logarithm.
- Implemented: Yes

IntendedITI

- Description: The intended ITI duration (s).
- Implemented: Yes

ITIReset

- Description: Whether the ITI partially resets if they try to poke in before it ends.
- Implemented: Yes

MaxWait

- Description: The maximum allowed time to start the trial (s).
- Implemented: Yes

UseRT

- Description: Whether the sound stops with the animal leaving the CNP.
- Implemented: Yes

UseMaxRT

- Description: Whether there is a MaxRT.
- Implemented: Yes

MaxMT

- Description: The maximum allowed time to move to the LNP (s).
- Implemented: Yes

AbortPenalty

- Description: The abort penalty time (s).
- Implemented: Yes

IncorrectPenalty

- Description: The incorrect answer penalty time (s).
- Implemented: Yes

FixationAbortPenalty

- Description: The fixation abort penalty time (s).
- Implemented: Yes

UsePerformance

- Description: Whether there is a minimum performance requirement to advance block.
- Implemented: Yes

CriticalPerformance

- Description: The minimum correct answer ratio required to advance block (if UsePerformance is 1).
- Implemented: Yes

MaxAborts

- Description:
- Implemented: No

RepeatError

- Description: Whether the stimulus is repeated after incorrect responses.
- Implemented: Yes

RepeatAbort

- Description: Whether the stimulus is repeated after aborts.
- Implemented: Yes

Speakers

- Description: Whether the animal is using headphones or not (Headphones = 1, Box Speakers = 0).
- Implemented: Yes

AbortLight

- Description:
- Implemented: No

ITILight

- Description:
- Implemented: No

4.3 Output File

The output file that comes out at the end of a session is also a CSV file. In this case, each row corresponds to a different trial. Since this is the file that contains the data that is going to be analyzed and since some trials may be aborted (and at different stages of the state machine), it may be convenient to understand what certain values for some output parameters mean:

- For the output parameters which correspond to a timed event (eg: trial_duration, ITI_end, timed_fix, etc.), if the a trial is aborted before one of those parameters is set, the value written in the output file is 0;
- For the response_poke output parameter, -1 corresponds to the left poke, 1 to the right poke and 0 to an abort.

Animal

- Description:
- Implemented: Yes

Session

- Description:
- Implemented: Yes

SessionType

- Description:
- Implemented: Yes

Trial

- Description:
- Implemented: Yes

Block

- Description:
- Implemented: Yes

TrialsPerBlock

- Description: Number of trials of the current block.
- Implemented: Yes

TrainingLevel

- Description:
- Implemented: Yes

ABL

- Description:
- Implemented: Yes

ILD

- Description:
- Implemented: Yes

Bias

- Description: Left < 0.5, Right > 0.5.
- Implemented: No

LeftAmp

- Description:
- Implemented: No

RightAmp

- Description:
- Implemented: No

WaveformL

- Description: The index of the sound played in the left speaker during the current trial.
- Implemented: No

WaveformR

- Description: The index of the sound played in the left speaker during the current trial.
- Implemented: No

TrialStart

- Description: Trial start timestamp (s).
- Implemented: Yes

TrialEnd

- Description: Trial end timestamp (s).
- Implemented: Yes

TrialDuration

- Description: Duration of the trial (s).
- Implemented: Yes

IntendedITI

- Description: The intended ITI duration (s).
- Implemented: Yes

ITIStart

- Description: ITI start timestamp (s).
- Implemented: Yes

ITIEnd

- Description: ITI end timestamp (s).
- Implemented: Yes

TimedITI

- Description: Duration of the ITI (s).
- Implemented: Yes

MaxWait

- Description: The maximum allowed time to start the trial (s).
- Implemented: Yes

TimeToCNP

- Description: The amount of time it took for the animal to poke in the CNP (s).
- Implemented: Yes

BaseFT

- Description: The constant part of the fixation time (ms).
- Implemented: Yes

ExpFTMean

- Description: The mean value of the random part of the fixation time (ms).
- Implemented: Yes

IntendedFT

- Description: The intended FT duration for the trial (ms).
- Implemented: Yes

TimedFT

- Description: The actual FT duration (s).
- Implemented: Yes

BaseRT

- Description: Minimum amount of time in CNP to wait after the sound presentation starts (s).
- Implemented: Yes

MaxRT

- Description: Maximum amount of time in CNP to wait after the sound presentation starts (s).
- Implemented: Yes

TimedRT

- Description: The amount of time the animal waited in the CNP after the sound presentation started (s).
- Implemented: Yes

MaxMT

- Description: The maximum allowed time to move to the LNP (s).
- Implemented: Yes

TimedMT

- Description: The amount of time it took for the animal to poke in a LNP (s).
- Implemented: Yes

IntendedLNP

- Description: Minimum poke duration in LNP (s).
- Implemented: Yes

TimedLNP

- Description: The amount of time the animal kept poking the LNP (s).
- Implemented: Yes

ResponsePoke

- Description: The side (Left = -1, Right = 1, Abort = 0) that the animal considered the sound was played the loudest.
- Implemented: Yes

Success

- Description: Whether the animal got the answered correctly (1) or not (0).
- Implemented: Yes

AbortEvent

- Description: Whether the trial aborted (1) or not (0).
- Implemented: Yes

RepeatTrial

- Description: Whether this trial should be repeated (1) or not (0).
- Implemented: Yes

BlockPerformance

- Description: The ratio of right answers to non-abort trials.
- Implemented: Yes

BlockAbort

- Description: The ratio of abort trials to the total number of trials.
- Implemented: Yes

LEDTrial

- Description:
- Implemented: No

TimedLED

- Description:
- Implemented: No

LEDPowerL

- Description:
- Implemented: No

LEDPowerR

- Description:
- Implemented: No

Box

- Description: The number of the box where the task was conducted.
- Implemented: Yes