LSL hands-on 2:

Latency tests and how to LSL on a heavy 3D environment

Tea Time 13.08.2020 - Doborah Nolte and Marc Vidal De Palol

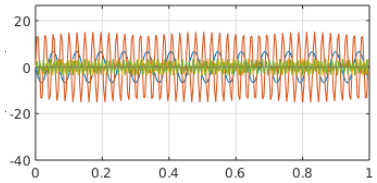
Outline

1. Recap: Aligning data streams sampled from different sources
2. Recap: Lab Streaming Layer (LSL) as data streams aligning solution
3. How to use LSL (Unity with EEG use case)
4. Test design
5. Recap: How does the data look like?
6. Results
7. Conclusions and remarks
8. Future outlook

1. Recap: Aligning data streams sampled from different sources

Data sampling devices have different:

- Sampling rates (e.g. 1000Hz for EEG, 60FPS in Unity, 90Hz for Eye-Tracking, etc.)
- CPU clocks (diff. CPU clocks between computers, EEG amplifiers, etc.)



2. Recap: Lab Streaming Layer (LSL) as data streams aligning solution

Why LSL?

- Manually aligning data streams can be really time consuming
- Standard tool in the science community: community support :)

Also:

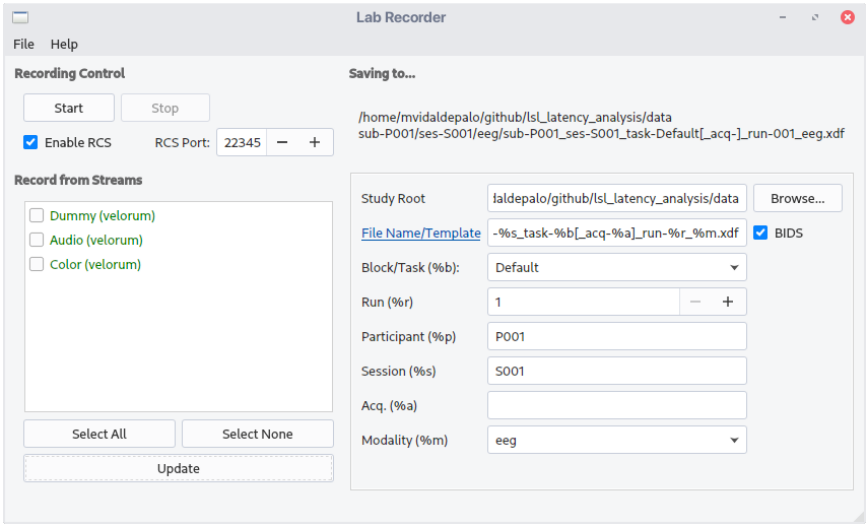
- open source
- cross platform (Win, Linux, MacOS, Android, iOS) and multi language APIs (C, C++, Python, Java, C#, MATLAB)
- many GUI tools supporting standard technologies and devices
- XDF data recordings (meant to be a standard format in science)

Source: https://labstreaminglayer.readthedocs.io/info/intro.html#what-is-lsl (https://labstreaminglayer.readthedocs.io/info/intro.html#what-is-lsl)

3. How to use LSL (Unity with EEG use case)

How to use LSL? General requirements:

- LabRecorder app: [https://github.com/labstreaminglayer/App-LabRecorder/releases (https://github.com/labstreaminglayer/App-LabRecorder/releases)](https://github.com/labstreaminglayer/App-LabRecorder/releases)



- `liblsl` library ([https://github.com/sccn/liblsl/releases/latest (https://github.com/sccn/liblsl/releases/latest)](https://github.com/sccn/liblsl/releases/latest)) with code defining your data streams

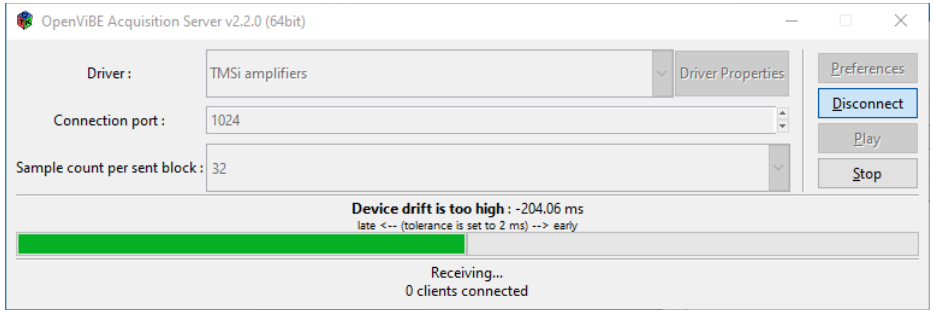
  and/or

3. How to use LSL (Unity with EEG use case)

How to use LSL? General requirements:

`and/or`

- LSL community app (e.g. OpenVibe Data Acquisition Server: [http://openvibe.inria.fr/downloads/ (http://openvibe.inria.fr/downloads/)](http://openvibe.inria.fr/downloads/)(http://openvibe.inria.fr/downloads/))



3. How to use LSL (Unity with EEG use case)

Code example in C# defining a data stream:

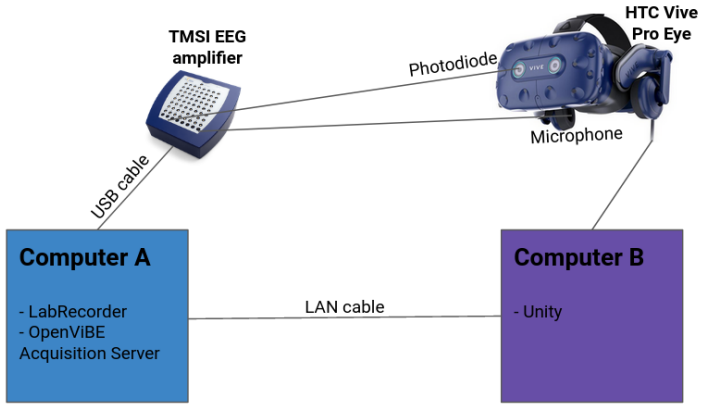
```
using LSL;
public class yourClass
{
    // declare your data stream
    private liblsl.StreamInfo lslStreamInfo; // stream information
    private liblsl.StreamOutlet lslOutlet; // actual stream to send out
    void startingMethod() { // normally Start() in Unity
        // instantiate your stream
        lslStreamInfo = new liblsl.StreamInfo(
            sName, // name of your stream
            sType, // stream type
            nValues, // you send an array, so how many values it contains
            nominalRate, // constant or not
            LslChannelFormat, // if you send integers, floats, etc.
            uuid); // universally unique identifier
        lslOutlet = new liblsl.StreamOutlet(streamInfo);
    }
    void sendingMethod() { // normally FixedUpdate() in Unity
        var data = new float[nValues]; // data you want to send
        lslOutlet.push_sample(data); // send/stream your data
    }
}
```

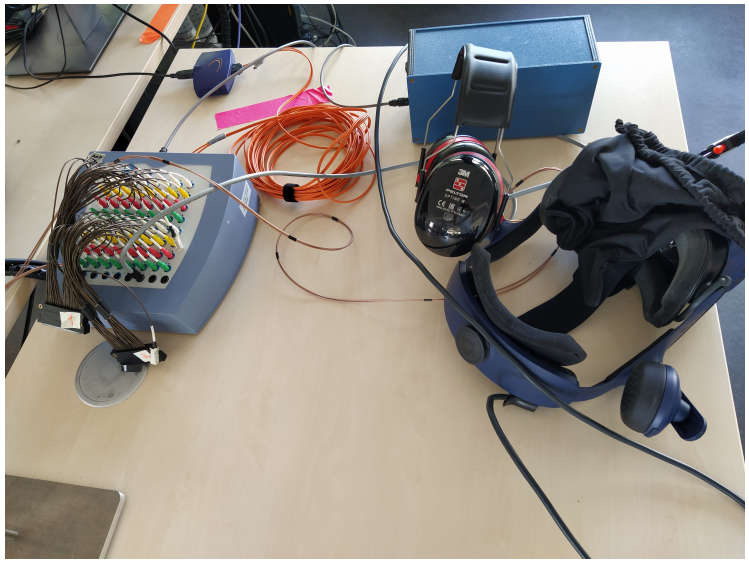
More examples https://labstreaminglayer.readthedocs.io/dev/examples.html (https://labstreaminglayer.readthedocs.io/dev/examples.html)

API reference: https://labstreaminglayer.readthedocs.io/projects/liblsl/index.html (https://labstreaminglayer.readthedocs.io/projects/liblsl/index.html)

3. How to use LSL (Unity with EEG use case)

Use case (our test setup):

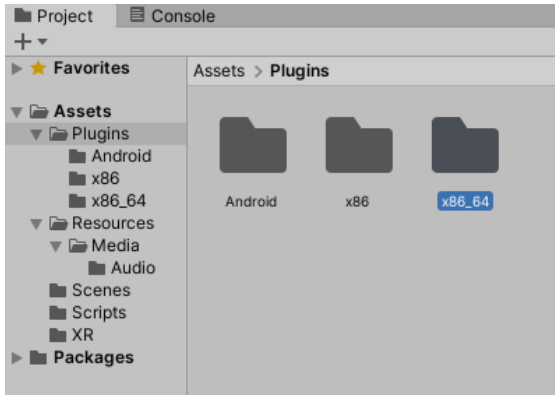




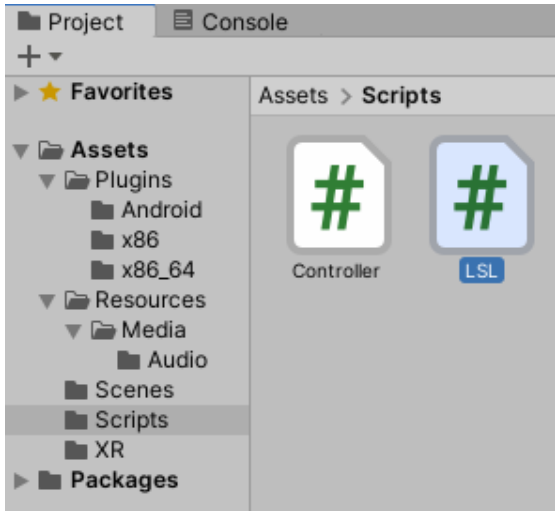
3. How to use LSL (Unity with EEG use case)

Where to place `liblsl` on Unity:

- `liblsl` library



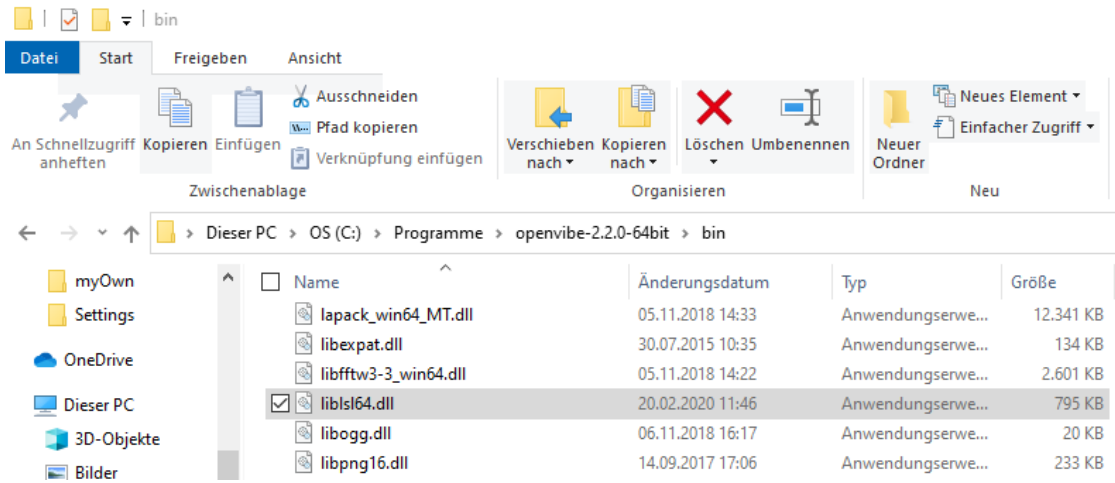
- `LSL.cs` class



3. How to use LSL (Unity with EEG use case)

How to make OpenVibe Acquisition Server (AVAS) work with `liblsl`:

- `liblsl` library replacement

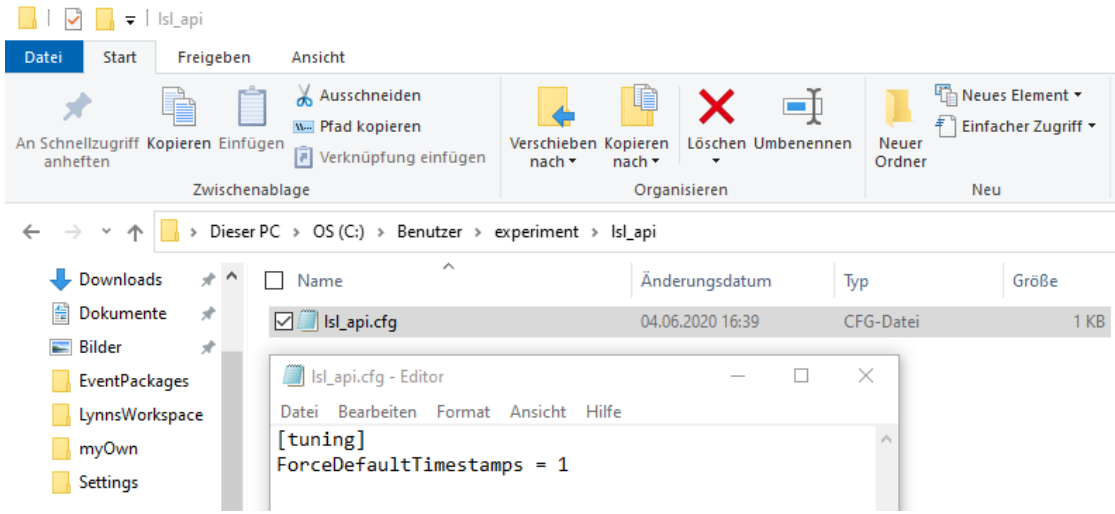


Important: make sure you share the same `liblsl` version between the LabRecorder, Unity and your scripts and/or any community LSL app. You can always overwrite the shipped one if you're unsure.

3. How to use LSL (Unity with EEG use case)

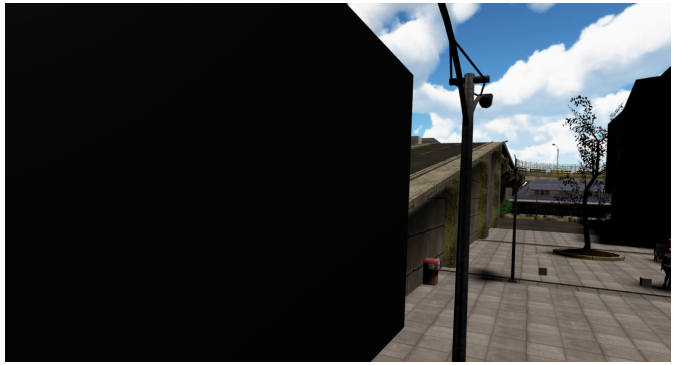
How to make OpenVibe Acquisition Server (AVAS) work with `liblsl`:

- LSL configuration file library "hack"



4. Test design

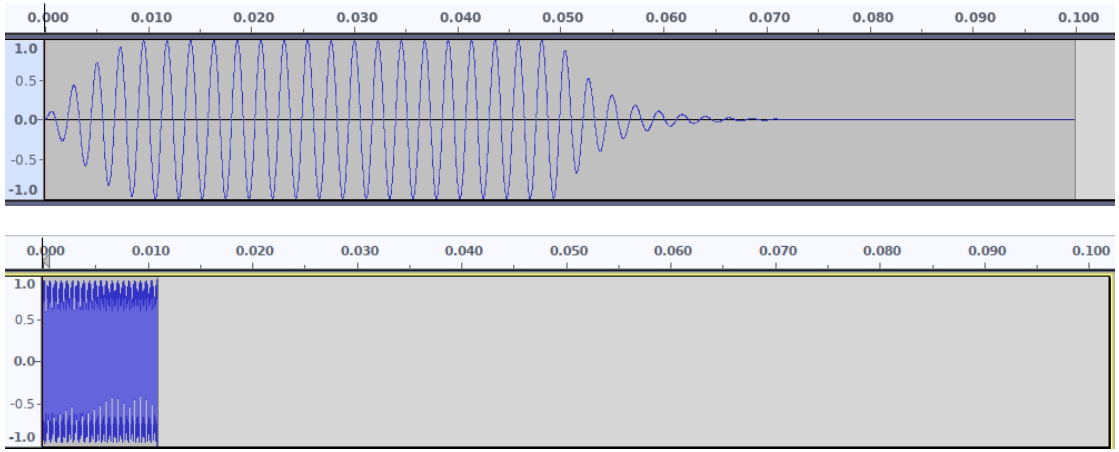
Background (2D vs 3D):



Unity 2D project used for testing: [https://github.com/mvidaldp/lsl_in_unity (https://github.com/mvidaldp/lsl_in_unity)](https://github.com/mvidaldp/lsl_in_unity)

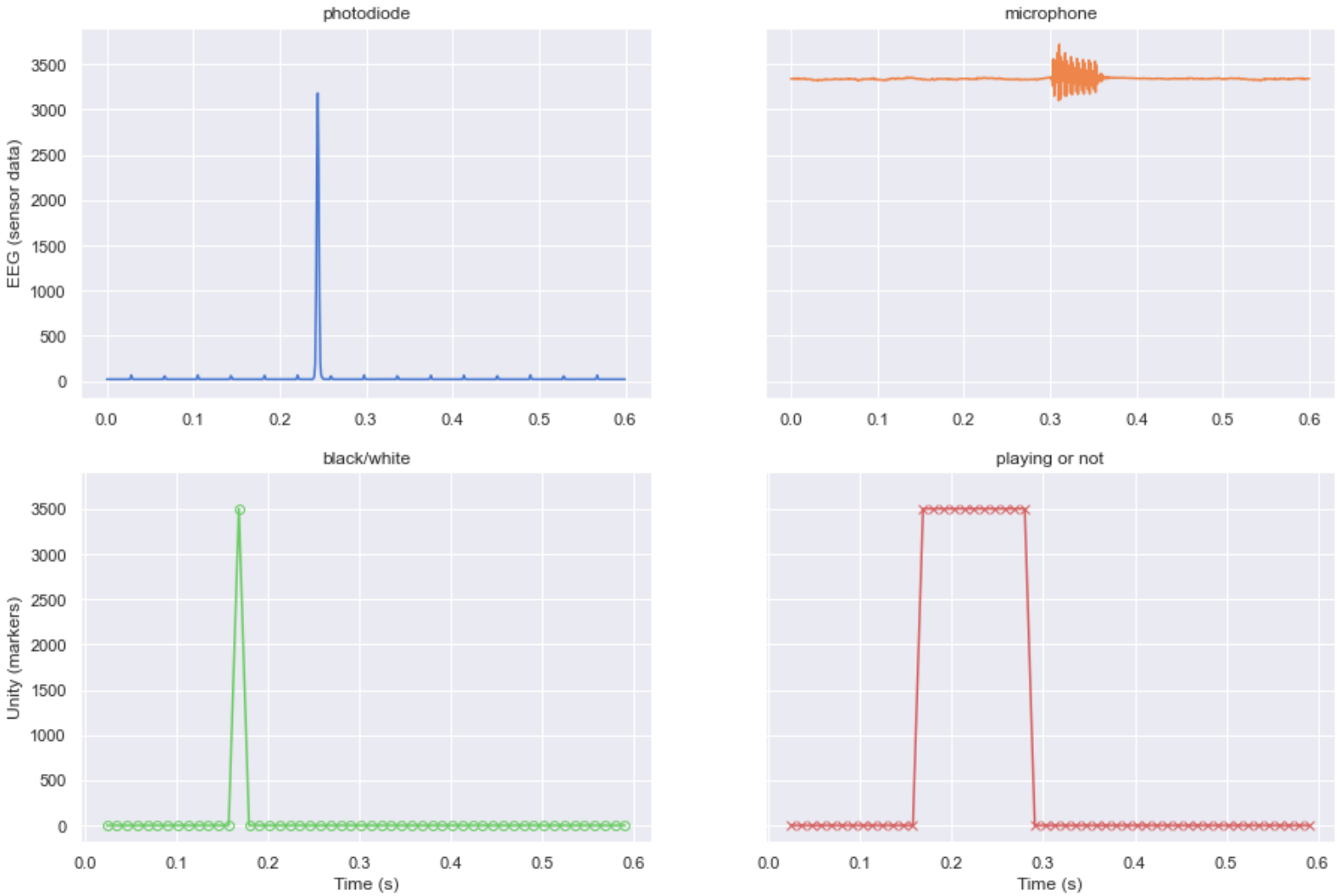
4. Test design

Audio (old vs new):

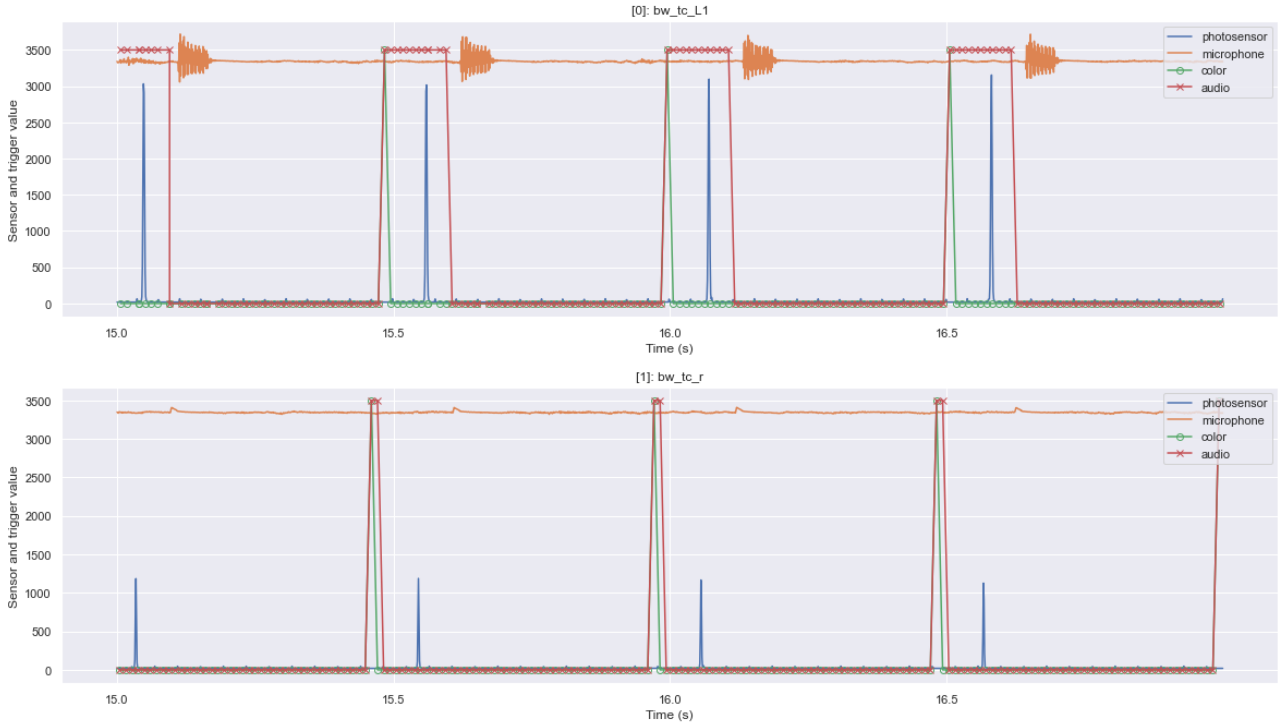


Unity 2D project used for testing: [https://github.com/mvidaldp/lsl_in_unity (https://github.com/mvidaldp/lsl_in_unity)](https://github.com/mvidaldp/lsl_in_unity)

5. Recap: How does the data look like?



5. Recap: How does the data look like?



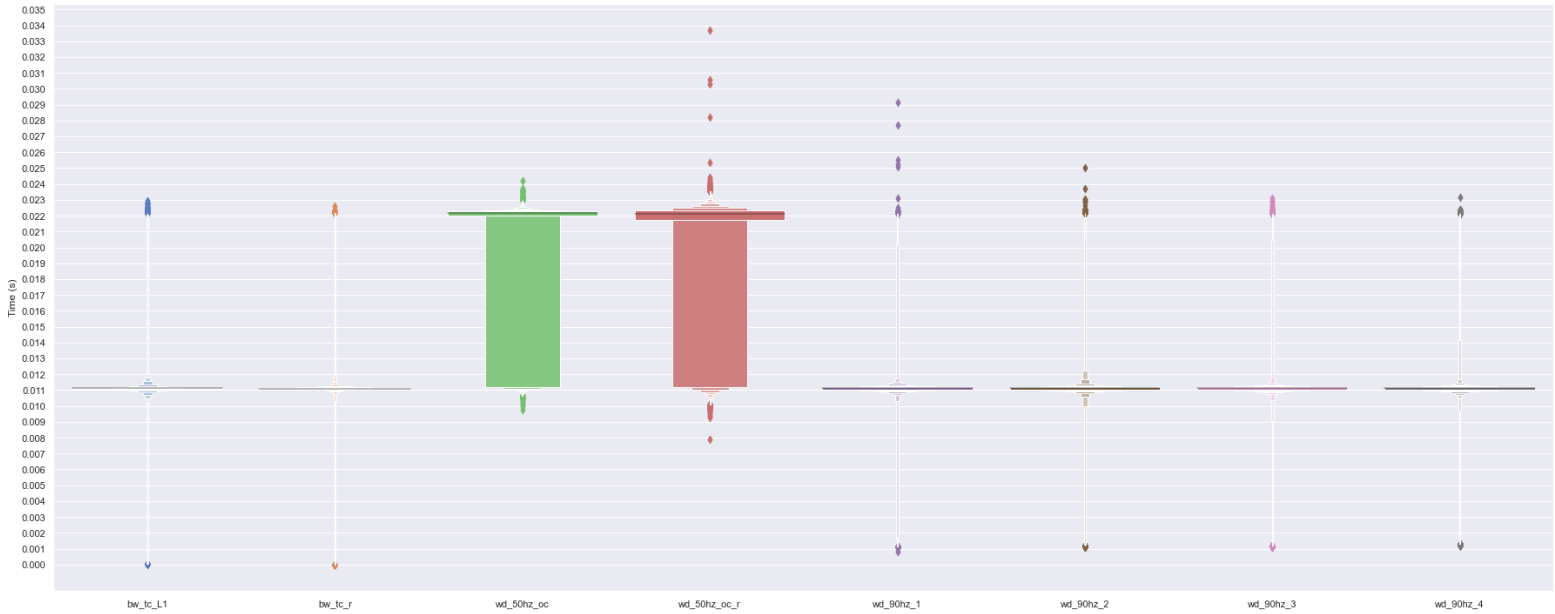
What do we look for exactly?

- **Image latency:** time between unity color markers and the diode peaks
- **Audio latency:** time between unity audio playing markers and microphone peaks

6. Results

How constant are the framerates?

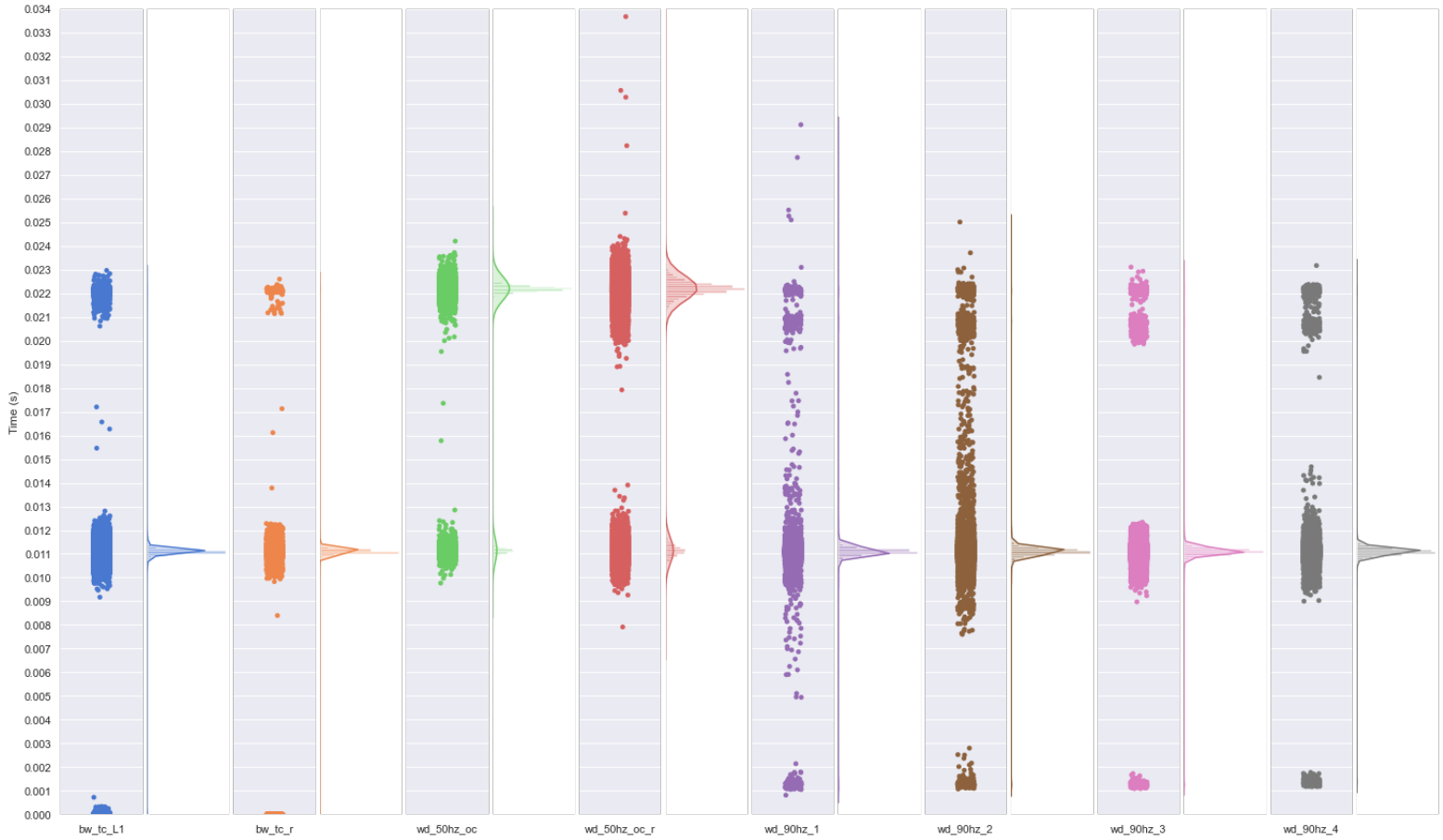
Unity (90FPS ~ 11.11ms vs 50FPS ~ 20ms)



6. Results

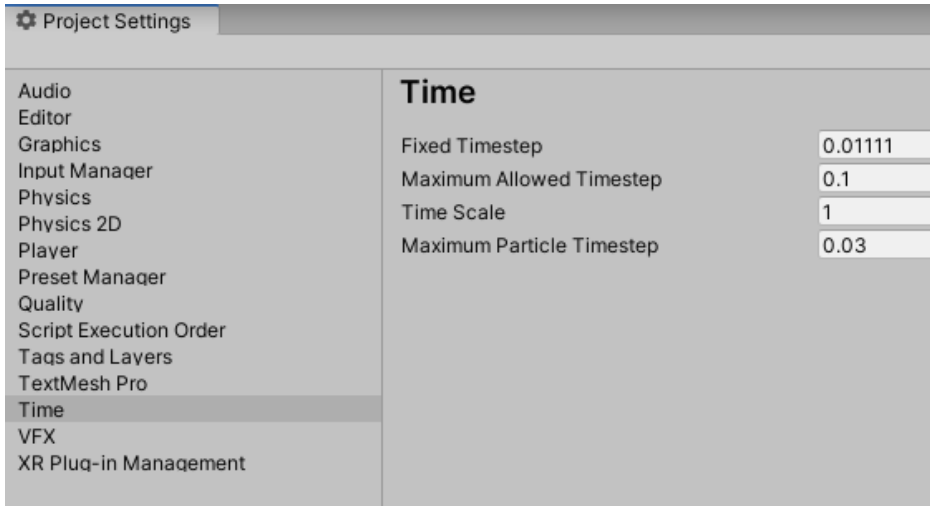
How constant are the framerates?

Unity (90FPS ~ 11.11ms vs 50FPS ~ 20ms)



6. Results

How to enforce 90FPS instead of 50FPS (default) in Unity?



Important: Although this parameter works, it makes Unity more prone to skip frames. Also, on the Unity Editor Player we could not run Westdrive on a `NVIDIA RTX 2080Ti` ! However, it worked on the build :)

6. Results

Latencies

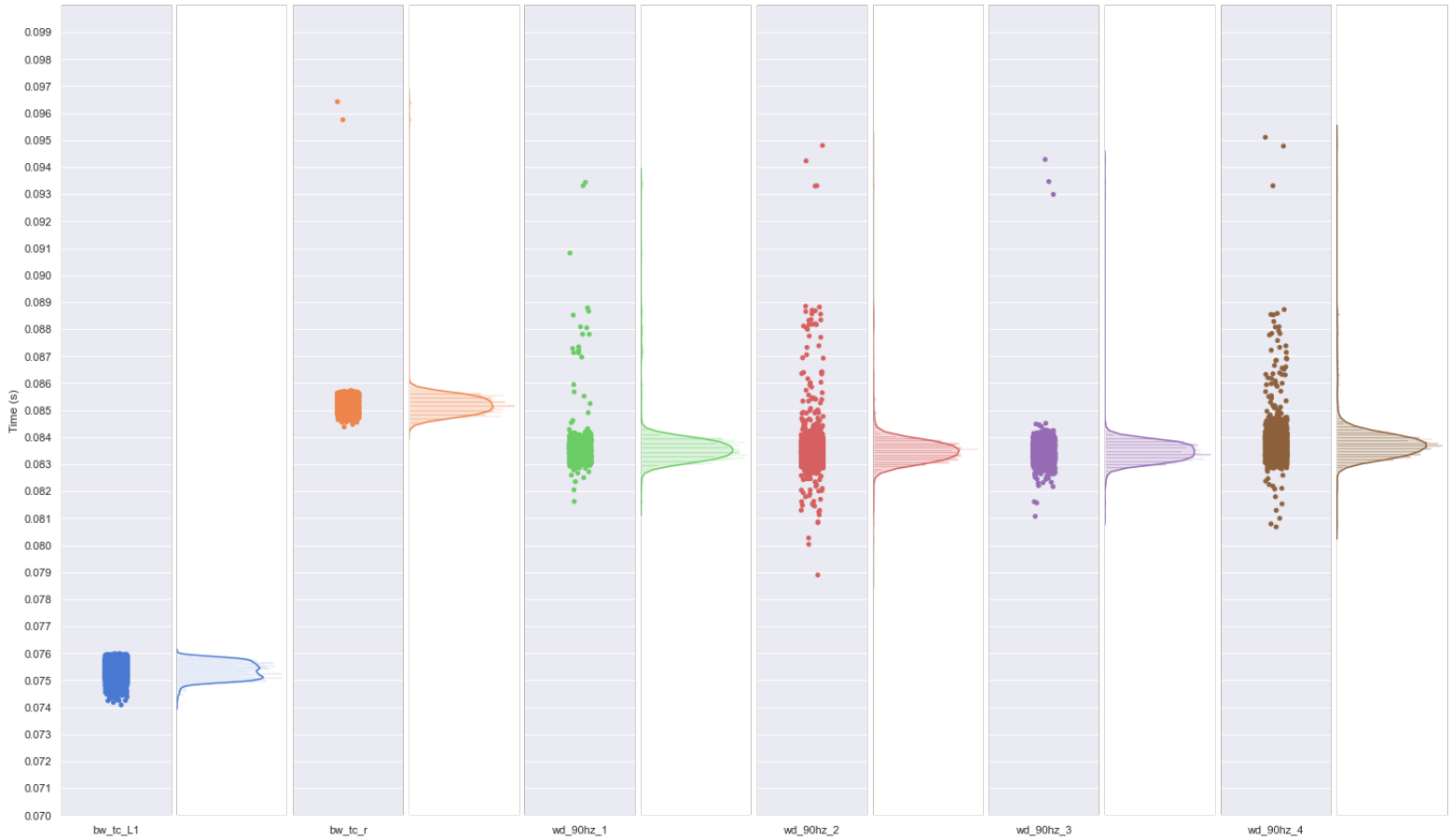
Time distance between unity color markers and the diode peaks

	bw_tc_L1	bw_tc_r	wd_90hz_1	wd_90hz_2	wd_90hz_3	wd_90hz_4	AVG
count	7706.000000	856.000000	1218.000000	2407.000000	2385.000000	2414.000000	2831.000000
mean	0.075361	0.085203	0.083596	0.083538	0.083470	0.083755	0.082487
std	0.000301	0.000601	0.000715	0.000785	0.000491	0.000703	0.000599
min	0.074084	0.084378	0.081619	0.078895	0.081069	0.080676	0.080120
25%	0.075118	0.084939	0.083269	0.083209	0.083205	0.083406	0.082191
50%	0.075364	0.085185	0.083534	0.083479	0.083460	0.083697	0.082453
75%	0.075610	0.085423	0.083796	0.083741	0.083716	0.083968	0.082709
max	0.075989	0.096419	0.093438	0.094801	0.094282	0.095103	0.091672

6. Results

Latencies

Time distance between unity color markers and the diode peaks



6. Results

Latencies

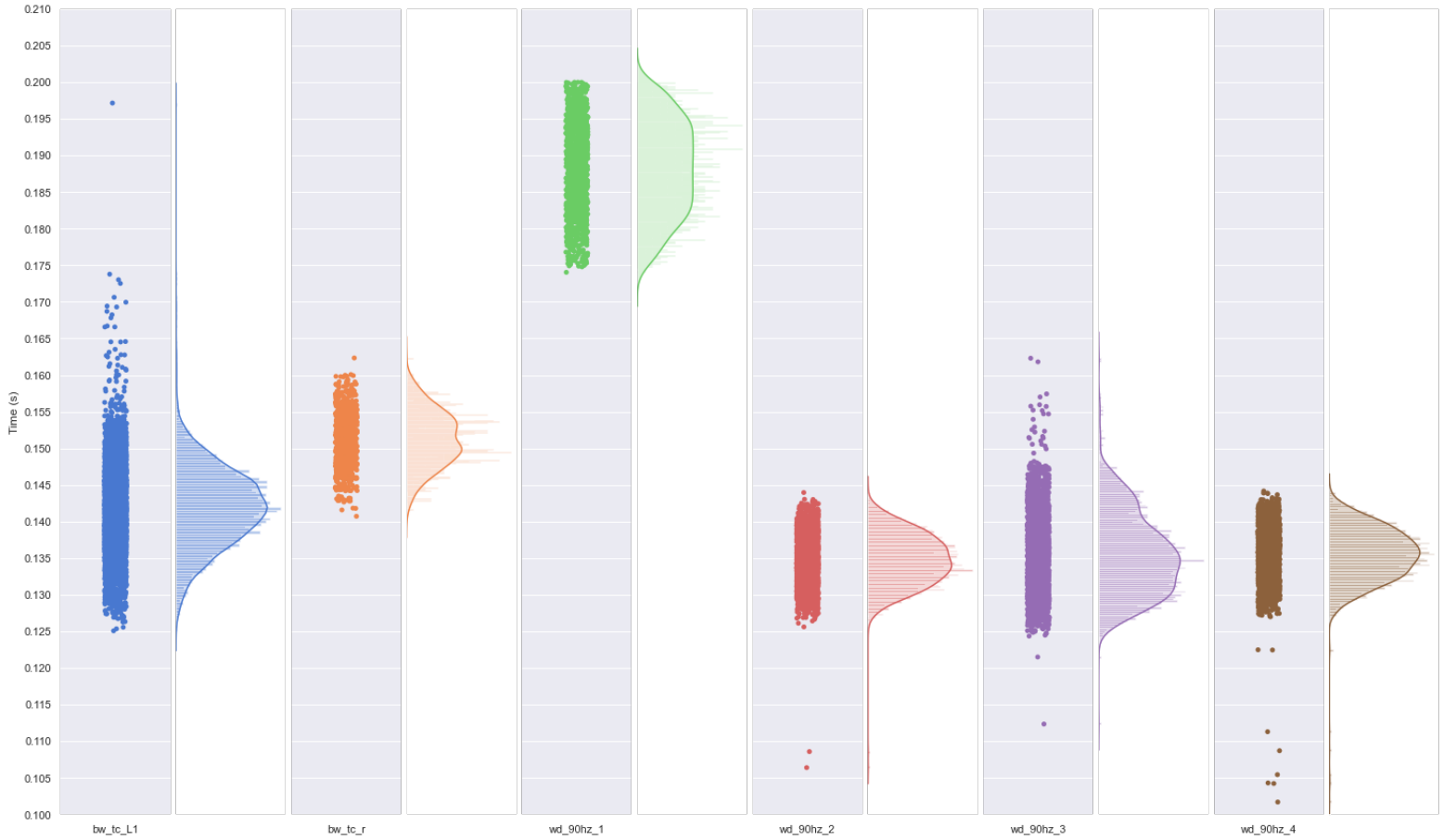
Time distance between unity audio playing markers and microphone peaks

	bw_tc_L1	bw_tc_r	wd_90hz_1	wd_90hz_2	wd_90hz_3	wd_90hz_4	AVG
count	7744.000000	860.000000	1185.000000	2416.000000	2395.000000	2423.000000	2837.166667
mean	0.141957	0.151279	0.188110	0.134765	0.135461	0.135430	0.147833
std	0.005408	0.003756	0.006367	0.003473	0.005617	0.003710	0.004722
min	0.125079	0.140721	0.174040	0.106402	0.112354	0.101700	0.126716
25%	0.138423	0.148477	0.183037	0.132218	0.131132	0.132934	0.144370
50%	0.141960	0.151101	0.188238	0.134754	0.134877	0.135556	0.147748
75%	0.145502	0.153893	0.193318	0.137348	0.139080	0.138031	0.151196
max	0.197155	0.162342	0.199990	0.143965	0.162306	0.144170	0.168321

6. Results

Latencies

Time distance between unity audio playing markers and microphone peaks



7. Conclusions and remarks

- Unity skips 1-2% of the triggers (frames) when enforcing it to 90FPS instead of the 50FPS (default)
- A better CPU on the recorder computer seems to contribute on a more precise aligning of streams
- Same goes for a better GPU to keep the framerate as constant as possible
- Latency for video looks pretty constant ~75ms and std of 0.4ms (2D old recordings), ~83ms and std of 0.4-0.7ms (2D and 3D new recordings)
- We assume this latency is project and device dependant. So, test your project to compute your latency!
- Latency for audio ~135ms or higher (std of 3ms-6ms), also project dependant
- Best latency for audio parameter seems to reduce it and also to reduce variance, but still far from usable for EEG (min-max diff > 10ms)
- **Using LSL for recording Unity and EEG data looks reliable for image stimuli**

8. Future outlook

- New recordings and data analysis using a our Westdrive scenes with the moving camera
- Redo the recordings in a sound isolated room to see if the audio latency detection improves
- Try a new audio engine if there is any that could help getting better results

Resources

- Liblsl (https://github.com/sccn/liblsl/releases/latest)
- LabRecorder (https://github.com/labstreaminglayer/App-LabRecorder/releases)
- LSL4Unity (https://github.com/labstreaminglayer/LSL4Unity)
- liblsl-Csharp (https://github.com/labstreaminglayer/liblsl-Csharp)
- LSL Apps (https://github.com/sccn/labstreaminglayer/tree/master/Apps)
- LSL documentation reference (https://labstreaminglayer.readthedocs.io/)
- LSL latency analysis (https://github.com/mvidaldp/lsl_latency_analysis)
- LSL in Unity (https://github.com/mvidaldp/lsl_in_unity)
- How to install and use LSL tutorial (https://docs.google.com/document/d/1NCLiLwMQpt9QKIw7sOM4N62vdDu3Nyw78928Jpz5VIU/edit?usp=sharing)
- OpenVibe (http://openvibe.inria.fr/downloads/)
- Audio tones generator (https://github.com/mvidaldp/pytonegen)