March 10, 2021

Executing the TactileMusic\_pause\_stop.py multiple times, shows that the computational time for an interval does change from one execution to next. The increase or decrease in computational time is not consistent for all segments. i.e. during the same run, one segment might need less time to compute while another one might need more time to compute – depending on the number of tasks running on the system and the priority of this task possibly.

Based on the trials ran and the worst case so far (~0.6 seconds total computational time), it looks like 1 sec time interval might work with ~30-40% buffer as of now. 1 second is a long time-interval.

We cannot expect that no other processes will be running on the given system or figure out a time interval each time based on the processing capability at that point. So, lets pick a random long interval – 0.5 seconds and skip sending the values for that segment if the computation took longer than 0.5 seconds. Yes, if the processor is really slow (like mine), it will skip a lot of segments but at least it will not miss the synch.

Another way to attempt synchronization may be to preprocess the entire audio file and send the motor values every 0.05 seconds – so that it is processor independent and works for short time intervals.

For making it completely processor independent, the preprocessed data with the motors to be activated could be flashed onto the ESP32. Probably not the best idea. Also there might be more desynch issues with the audio from one device and motor activation from another.

-----------------------------------------------------------------------------------------------------------

Playing **test.wav**

**interval = 0.05**

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.49304962158203125

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.47985363006591797

-------

Segment: 0

Time Slept: 0.000000

Sleep Needed: -0.479854

Data Intensity: 2003.615387

--------

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.5177500247955322

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.514293909072876

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.520409107208252

**interval = 0.5**

Segment: 0

Sleep Needed: -0.056091

Data Intensity: 1591.468168

Playing **spoopy.wav**

**interval = 0.05**

Segment: 0

Sleep Needed: -0.477311

Data Intensity: 3.490677

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.4928734302520752

segment: 368

sleep needed: -0.005112171173095703

segment: 714

sleep needed: -0.29038453102111816

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.48284244537353516

segment: 919

sleep needed: -0.0023517608642578125

----- processing time exceeded time interval -------

segment: 108

sleep needed: -0.058211326599121094

segment: 109

sleep needed: -0.012827396392822266

segment: 124

sleep needed: -0.002827167510986328

segment: 895

sleep needed: -0.0002620220184326172

**Processor load - CPU: ~5%, Memory ~45%**

#After restarting the system, although zoom was running.

**Processor load - CPU: ~30%, Memory: ~45%**

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.04213094711303711

segment: 259

sleep needed: -0.008948326110839844

segment: 260

sleep needed: -0.04009246826171875

segment: 261

sleep needed: -0.03344011306762695

……… multiple segments popped up when this word document was opened.

segment: 573

sleep needed: -0.0019466876983642578

segment: 921

sleep needed: -0.010659933090209961

segment: 976

sleep needed: -0.0029473304748535156

segment: 987

sleep needed: -0.017937421798706055

----- processing time exceeded time interval -------

segment: 0

sleep needed: -0.055747270584106445

segment: 293

sleep needed: -0.0008180141448974609

segment: 694

sleep needed: -0.03775930404663086

-------------------------------------------------------------------