Documentation for demo
Distributed Synchronization for Ad-Hoc Acoustic Sensor Networks
(Using Closed-Loop Double-Cross-Correlation Processing)

Niklas Knäpper, Aleksej Chinaev, Gerald Enzner

24.08.2022

With the concept of signal processing in wireless acoustic sensor networks, network-wide synchronization is required to exhaust their potential in signal enhancement. Our demo presents a gossiping approach for distributed synchronization using waveform-based estimation of sampling rate offset (SRO) by the double-cross-correlation processor (DXCP) [1]. To this end, a Jupyter notebook is implemented on asynchronous microphone signals, which are simulated by using an acoustic sensor network signal generator (ASNSI), which makes use of the room impulse responses of simulated sound interface to the swarm (SINS) environment and is available at https://github.com/fgnt/asnsig with some source code examples.

Contents

1 Installation

1.1 Demo software and its dependencies

The software for this demo can be downloaded either from

- https://cloud.uol.de/s/gQ7C9AcQzkjT8n4

or obtained directly from the GitHub-repository https://github.com/fgnt/asnsig.

In order to run this demo, make sure to install the following dependencies first: jupyter (1.0.0), numpy (1.21.0), numpy-stl (2.16.3), scipy (1.8.0), matplotlib (3.5.1), tqdm (4.62.3) and a Python package asnsig.

1.2 Required data

The required data consist of a stl-file containing a geometry of the simulated SINS environment, two json-files (positions.json and testbed.json) and a set of audio signals provided in several wav-files. The data can be downloaded for different signal length from

- https://cloud.uol.de/s/WHjb4RoAEo68dFT (of $\simeq$ 96 MB for 150 seconds)
- https://cloud.uol.de/s/7FmcpgTrePXM7Jd (of $\simeq$ 190 MB for 300 seconds)
- https://cloud.uol.de/s/BNLg56cgLZZawMc (of $\simeq$ 1.1 GB for 1800 seconds)

and placed inside the **data/** directory of the demo software. If you choose to use another directory, make sure to adjust the `DATA_ROOT` parameter inside the notebook accordingly. Note, the content of the data provided for this demo does not correspond to the data used in [1]. Further, the audio signals of every downloaded zip-file contain its own sensor noise different from other data sets.

2 Configuration

2.1 Simulated network topology

Topologies are configured as a directed tree and defined by their levels in *nodes_levels*

```
nodes_levels = [level0, level1, ...].
```

Each *level*, apart from the very first, may contain multiple *branches*

```
level1 = [branch0, branch1, ...],
```

where one branch describes a root-node and a set of leaf-nodes connected to it. For example, in the branch

```
branch0 = ['node_0', 'node_1', 'node_2'],
```

node_1 and *node_2* are connected to *node_0*, meaning they receive it's synchronized signal as a reference. Notice that for any node to be a valid root, it must either be within the first level, making it the root of the entire tree, or appear as a leaf in the preceding level. Please refer to [1] for more details.

The simulated acoustic sensor network includes 13 nodes, indexed from 0 to 12. The following code provides examples for each of the proposed topologies:

a) Example for star out-tree (SOT)

```
nodes_levels = [
    [['node_0', 'node_5', 'node_2', 'node_3']]
]
```

b) Example for path out-tree (POT)

```
nodes_levels = [
    [['node_0', 'node_5']],
    [['node_5', 'node_3']],
    [['node_3, 'node_8']]
]
```

c) Example for rooted out-tree (ROT)

```
nodes_levels = [
    [['node_0', 'node_4', 'node_5']],
    [['node_4', 'node_6', 'node_7'], ['node_5, 'node_8']],
    [['node_7', 'node_10', 'node_11']]
]
```

2.2 Data directory, signal duration and SSNR calculation

- If you decided not to use the default database directory, please adjust the database path in `DATA_ROOT`.
- Use `sig_len_sec` to set the signal duration in seconds. Please be aware of the maximum duration for either the small or large database.
- The variable `ssnr_t_max` controls the time in seconds up until which the signal-to-synchronization-noise ratio (SSNR) is calculated, starting only after SRO estimation at the respective node has converged as described in [1]. Due to the methods employed, SSNR results are influenced significantly by this parameter. By default, it is set to the signal duration. Adjustments are useful if your simulation includes periods of slightly higher errors in SRO estimation after the initial convergence, especially if it is at the end.

2.3 Flags for different simulation modes

- *Online/offline simulation*: Use the flag `SIMULATE_ONLINE` to toggle between two available simulation modes. If set to `True`, SRO estimation and synchronization is simulated on a per-frame basis, where in each time step, every node processes an input- and generates an output-frame. Therefore, the processing pattern closely resembles that of a real network. If set to `False`, simulation is carried out in an optimized manner, utilizing parallel processing where possible. Depending on the configured topology, this may lead to significantly reduced execution times. In any case, the expected results are identical for both modes.

- *Starting control of SRO estimation*: Since nodes may require an extended time span for initialisation depending on their position within the tree, it is sometimes possible to reduce the overall SRO estimation error by freezing the SRO estimate until the incoming reference signal can be assumed to be synchronized reasonably well. Set `CONTROL_INIT_SRO_EST` to `True` to enable this functionality. However, please be advised that currently, this is an experimental feature.

3 Plots and evaluation results

3.1 Geometry of setup and configured topology

Figure 1 shows a geometry of the setup and the configured topology. Only sources that are active during the specified duration are displayed.

3.2 Source activity and scene diary

Figure 2 consists of two subfigures:

a) simulated source activity depicted over time separated by source-position;

b) scene diary describing overall character of acoustic scene (Dialogue, Music, etc.).

3.3 Running SRO estimation and signal synchronization

A progress of signal processing is depicted via *tqdm* progress bar with additional information such a code execution time, an estimated time for the code to complete etc..

3.4 Reference signal and average node SNR

Figure 3 consists of two subfigures:

- a microphone signal of the global root node. This serves as a visual guide. Of course, microphone signals of other nodes differ slightly;

- bar plot of node-specific SNR values on every node averaged globally over time assuming a constant noise-floor.

The resulting SNR values are also printed inline.

3.5 Resulting SRO estimation over time

Figure 4 consists of three subfigures:

a) SRO estimates for each node;

b) residual SRO estimates for each node. Since the residual SRO is zero for synchronized signals, it also indicates the synchronization success, but only when the acoustic conditions allow for a precise SRO estimation (For example, not during prolonged silence);

c) Time varying RMSE values (of SRO estimation) obtained via recursive averaging for each node.

3.6 Synchronization performance

Figure 5 displays the SSNR for each node (before and after synchronization) as a global evaluation metric for synchronization. The resulting SSNR values are also printed inline.

References

[1] A. Chinaev and G. Enzner, "Distributed Synchronization for Ad-Hoc Acoustic Sensor Networks Using Closed-Loop Double-Cross-Correlation Processing", *Int. Workshop on Acoustic Signal Enhancement (IWAENC 2022)*, Sept. 2022.