AUDIO SYNCHRONIZATION USING AUXILIARY-FUNCTION-BASED TIME DELAY ESTIMATION

Technical Report

Kouei Yamaoka

The University of Tokyo
kouei_yamaoka@ipc.i.u-tokyo.ac.jp

ABSTRACT

This report presents AuxSync, a non-learning system for synchronizing two independently recorded audio signals. The system estimates the parameters of an asynchronous signal model using cross correlation functions (CCFs). It first estimates the global sampling time offset using the CCF, whose peak is refined at sub-sample resolution using auxiliary-function-based time delay estimation. A double-cross-correlation processor is then used to estimate the global sampling rate offset. For datasets with reliable correlation structures, the global estimates are further refined around each keypoint; otherwise, a simpler global prediction is used to avoid unstable local corrections. On the validation data, the best submitted AuxSync variant achieved $0.376\,\mathrm{ms}$ mean absolute error for ARU recordings and $1113.8\,\mathrm{ms}$ for zebra finch recordings. These results show that classical signal-processing methods can provide accurate synchronization when the two channels share a clear acoustic structure, while sparse and repetitive vocalizations in noisy environments remain challenging.

***Index Terms*—** audio synchronization, time delay estimation, sample-rate offset, DXCP, AuxTDE

1. INTRODUCTION

Multichannel bioacoustic recording is important for studying animal behavior in the field. If recordings from independent devices can be synchronized after deployment, researchers can use larger and more flexible sensor layouts than those possible with a single device with a shared clock. Such recordings can support analyses of animal positions, group size, and communication dynamics, but clock drift between devices reduces the usability of the data unless it is corrected.

Supervised synchronization may be effective when enough annotated examples are available. However, collecting such annotations already requires some way to relate events across independently recorded channels. Relying only on supervised learning therefore creates a circular dependency: aligned data are needed to train the aligner that would make large-scale aligned data collection practical. This motivates synchronization algorithms that can work from the audio signals themselves, or from only a small amount of development data. The purpose of this report is to evaluate how far a non-machine-learning signal-processing system can improve alignment performance on this task.

2. SYSTEM

2.1. Problem Formulation

The goal is to predict, for each channel-0 keypoint sampled at one-second intervals, the corresponding timestamp in channel 1. Let $\tilde{x}_0(t)$ and $\tilde{x}_1(t)$ denote the underlying continuous-time signals observed by the two devices. Following the time-varying SRO model [1], and using channel 0 as the reference, the discrete observations can be modeled as

$$\begin{aligned} x_0[n] &= \tilde{x}_0(n/f_s), \\ x_1[n] &= \tilde{x}_1\left(\sum_{\nu=0}^{n-1} \frac{1}{(1+\epsilon[\nu])f_s} + \tau\right), \end{aligned} \tag{1}$$

where f_s is the sampling frequency, τ is the continuous sampling time offset (STO), and $\epsilon[\nu]$ is the sampling rate offset (SRO). We decompose it as

$$\epsilon[\nu] = \epsilon_{\mathrm{global}} + \epsilon_{\mathrm{tv}}[\nu], \tag{2}$$

where $\epsilon_{\mathrm{global}}$ is a file-level clock mismatch and $\epsilon_{\mathrm{tv}}[\nu]$ represents the time-varying residual. If the SRO is constant, (1) reduces to [2, 3]

$$x_1[n] = \tilde{x}_1\left(\frac{n}{(1+\epsilon)\,f_s} + \tau\right). \tag{3}$$

Estimating τ and ϵ yields synchronization between the two recordings. For submission, this is presented as the delay function

$$d(t) = t_1 - t_0, \tag{4}$$

where t_0 is a channel-0 keypoint and t_1 is the corresponding predicted channel-1 timestamp.

2.2. System Description

The system has three stages. It estimates the global STO τ, the global SRO component $\epsilon_{\mathrm{global}}$, and optionally estimates the local residual delay at each keypoint. The last stage should not be interpreted as a direct estimate of $\epsilon_{\mathrm{tv}}[\nu]$ itself; rather, it compensates for the delay residual that may be induced by time-varying SRO and by non-ideal local correlations. The PHAT (phase-transform) normalization used in the baseline method is not used in the submitted configuration.

2.3. Preprocessing

Before synchronization estimation, both channels are filtered with a zero-phase fourth-order Butterworth high-pass filter. The cutoff frequency is selected per dataset to suppress low-frequency noise that otherwise creates broad and uninformative correlation peaks.

Table 1: Main hyperparameters used by the submitted system.

Parameter	ARU	Zebra finch
High-pass cutoff frequency [Hz]	50	100
`max_error` [s]	0.5	3.0
`dxcp_sro_window_size` [s]	20.0	20.0
`dxcp_keypoint_window_size` [s]	1.0	N/A
`auxtde_n_iter`	50	50
Local refinement	true	false

2.4. Global STO Estimation

The first stage estimates the global recording-start offset τ. A whole-file cross correlation function (CCF) between the two channels within the allowed delay range is computed and its peak is refined by auxiliary-function-based time delay estimation (AuxTDE) [4]. This gives a sub-sample estimate of the global STO $\hat{\tau}$.

2.5. Global SRO

The second stage estimates $\epsilon_{\mathrm{global}}$. We use double-cross-correlation processor (DXCP), which estimates SRO from the drift of frame-wise CCFs. The method first computes primary CCFs between windowed frames and then computes secondary CCFs between primary CCFs at different frames [5,6]. In the submitted system, the peak of the secondary CCF is refined using AuxTDE as in [7]. The refined peak is then converted into the estimate $\hat{\epsilon}_{\mathrm{global}}$. The search range is determined from the maximum possible accumulated drift over one analysis frame.

2.6. Prediction and Local Refinement

The global estimates enable the prediction

$$d_{\mathrm{global}}(t) = \hat{\epsilon}_{\mathrm{global}} t - \hat{\tau}. \quad (5)$$

For datasets where local refinement is disabled, this function is used directly. Otherwise, AuxTDE is applied again around each keypoint. The local CCF is computed from a window whose length is set by `dxcp_keypoint_window_size`. The local residual delay is used as a keypoint-wise correction to the global estimates, not as an explicit estimate of $\epsilon_{\mathrm{tv}}[\nu]$. The final delay is clipped to $[-d_{\max}, d_{\max}]$, where $d_{\max}$ denotes the clipping threshold. Note that, for zebra finch, $d_{\max} = 3\,\mathrm{s}$ is smaller than the official possible desynchronization range of $\pm\,5\,\mathrm{s}$. This allows for an unpredictable error of up to $2\,\mathrm{s}$ for zebra finch. The predicted keypoints are finally clipped to the valid audio interval $[0, \texttt{audio\_dur}]$.

2.7. Optional outlier refinement

We also prepared submission variants with an additional label-free refinement step for outliers. This step is applied only to recordings whose predicted delay trajectory is flagged as unreliable, while all other data are left unchanged. A recording is flagged when two label-free signals agree: (i) the CCF used for the global STO estimate has no single dominant peak, measured as the ratio of its highest and second-highest peak, and (ii) the predicted delay trajectory itself contains an abnormally large frame-to-frame jump. Both quantities are computed from the audio signals and from the system's own predictions, without access to ground truth. A recording is flagged only when both signals indicate a problem.

Two correction strategies are considered for each flagged recording. First, if the delay trajectory is much better explained by two affine segments than by one, the audio is split at the detected changepoint and the global STO, SRO, and local residual estimation stages are re-run independently on the two segments. This is a heuristic fallback for recordings that appear to contain a clock discontinuity. Second, if isolated keypoints show abnormal frame-to-frame delay jumps, they are replaced by the linearly interpolated points using the adjacent normal delays. The abnormal jumps are detected by Hampel-style evaluation using median and median absolute deviation (MAD).

Neither correction is applied unconditionally. For each flagged recording, a robust roughness score is computed from the delay sequence using MAD, both for the original prediction and for each candidate correction; whichever of the two strategies and original prediction has the lowest score is kept. This allows the refinement to be run without ground-truth labels, but the procedure remains a heuristic exception path rather than part of the core estimator.

Table 2: Submission variants evaluated in this report.

System	Step 2 SRO	Outlier refinement
full	yes	no
full+refine	yes	yes
no-step2+sel-refine	no	ARU only

2.8. Submission variants

We evaluate three system variants. The first, *full*, is the direct implementation of the proposed pipeline described above. The second, *full+refine*, applies the optional outlier refinement after the full system. This variant is intended to test whether label-free post-processing can remove isolated failures without changing the core estimator. The third variant, *no-step2+sel-refine*, disables the global SRO estimation stage by setting $\hat{\epsilon} = 0$. For ARU, the optional outlier refinement is applied after this no-step2 estimate. For zebra finch, refinement is not applied because the correlation between the two signals is highly unreliable. This variant is included as a performance-oriented alternative when DXCP-based SRO estimation is less stable than the global STO plus local correction.

3. RESULTS AND DISCUSSION

3.1. Experimental Setup

We evaluated on the provided validation splits using the official evaluation script. The main metric reported here is mean absolute error (MAE) in milliseconds; mean squared error (MSE) is also reported in seconds squared. The submitted configurations use dataset-specific hyperparameters listed in Table 1. All parameters were determined independently of the evaluation dataset. The three submission variants are summarized in Table 2.

3.2. Quantitative Results

Table 3 summarizes validation results in the same format as the task description. Baseline results are copied from the challenge page,

Table 3: Validation-set performance. Lower is better.

	ARU		Zebra finch	
Model	MAE [ms]	MSE [s^2]	MAE [ms]	MSE [s^2]
nosync	116.2	0.019	2297.4	5.734
gccphat	122.4	0.020	2457.0	7.995
deeplearning	164.3	0.040	757.1	1.218
AuxSync full	0.656	8.71×10^{-5}	1255.3	3.703
AuxSync full+refine	0.376	4.03×10^{-5}	1191.1	3.119
AuxSync no-step2+sel-refine	0.376	4.03×10^{-5}	1113.8	3.657

Table 4: Train/validation performance of the system variants. Lower is better.

Dataset	System	Train MAE [ms]	Train MSE [s^2]	Val MAE [ms]	Val MSE [s^2]
ARU	full	1.286	2.04×10^{-4}	0.656	8.71×10^{-5}
ARU	full+refine	0.648	7.47×10^{-5}	0.376	4.03×10^{-5}
ARU	no-step2+sel-refine	0.559	7.15×10^{-5}	0.376	4.03×10^{-5}
Zebra finch	full	1612.3	5.018	1255.3	3.703
Zebra finch	full+refine	1644.2	5.056	1191.1	3.119
Zebra finch	no-step2+sel-refine	1428.5	4.718	1113.8	3.657

and the last three rows report the submitted AuxSync variants. Table 4 shows additional train/validation results used for the technical report.

3.3. Correlation Analysis

Figure 1 shows example CCF maps from the validation data. In the ARU example, the primary correlation contains a consistent high-energy ridge across frames, and the secondary correlation also has a localized peak region. This structure is compatible with the DXCP assumption that the delay peak drifts approximately linearly from frame to frame. Therefore, the global SRO can be estimated reliably and then refined locally by AuxTDE.

In contrast, the zebra finch example shows many isolated vertical responses and no stable peak in the primary CCF. The secondary CCF is also diffuse, possibly reflecting the sparsity and repeated nature of short vocalizations. This suggests that, for many zebra finch files, the available correlation structure is insufficient for reliable local keypoint refinement. We therefore disable it.

3.4. Discussion

For ARU, the optional outlier refinement substantially improves both train and validation performance. Disabling Step 2 gives a further small improvement on the training split and almost identical validation performance, suggesting that local AuxTDE refinement and outlier removal already absorb most residual drift in the current ARU dataset.

For zebra finch, disabling Step 2 gives the best validation MAE. This is consistent with the correlation analysis: when the secondary CCF is diffuse, DXCP-based global SRO estimation can introduce an incorrect drift. The full+refine system improves zebra finch validation MSE but is less stable across train/validation splits, so the no-step2 selective variant is used as the performance-oriented submission.

4. CONCLUSION

This report presented AuxSync, a non-learning synchronization system based on AuxTDE and DXCP. The results show that auxiliary-function-based delay refinement is highly effective for ARU recordings, where stable correlation structure is available. For zebra finch recordings, weak and scattered correlations make SRO estimation and local refinement less reliable; in this case, a simpler global-delay-based variant performed better. These results suggest that classical signal-processing methods can provide strong synchronization accuracy when the correlation structure is informative, but robust handling of sparse vocalizations remains the main limitation.

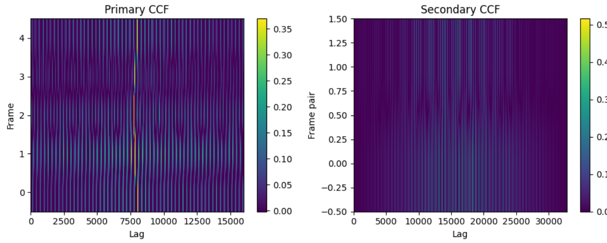
5. REFERENCES

[1] T. Gburrek, J. Schmalenstroeer, and R. Haeb-Umbach, “On synchronization of wireless acoustic sensor networks in the presence of time-varying sampling rate offsets and speaker changes,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2022, pp. 916–920.

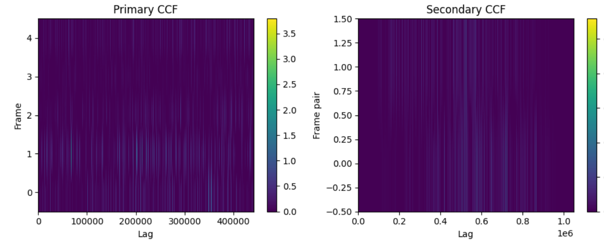
[2] S. Miyabe, N. Ono, and S. Makino, “Blind compensation of interchannel sampling frequency mismatch for ad hoc microphone array based on maximum likelihood estimation,” *Signal Process.*, vol. 107, pp. 185–196, 2 2015.

[3] L. Wang and S. Doclo, “Correlation maximization-based sampling rate offset estimation for distributed microphone arrays,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 24, no. 3, pp. 571–582, 3 2016.

[4] K. Yamaoka, R. Scheibler, N. Ono, and Y. Wakabayashi, “Sub-sample time delay estimation via auxiliary-function-based iterative updates,” in *Proc. IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2019, pp. 125–129.



(a) ARU



(b) Zebra finch

Figure 1: Example of primary and secondary CCF maps. The ARU example contains a usable correlation structure for DXCP, whereas the zebra finch example has weak and scattered correlation peaks.

[5] A. Chinaev, P. Thüne, and G. Enzner, “A double-cross-correlation processor for blind sampling rate offset estimation in acoustic sensor networks,” in *Proc. IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 641–645.

[6] A. Chinaev, P. Thüne, and G. Enzner, “Double-cross-correlation processing for blind sampling-rate and time-offset estimation,” *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 1881–1896, 4 2021.

[7] K. Yamaoka, N. Ono, and Y. Wakabayashi, “Sampling frequency mismatch estimation by auxiliary-function-based iterative maximization of double-cross-correlation,” in *Proc. European Signal Processing Conference (EUSIPCO)*, 2021, pp. 1125–1129.