Efficient Sampling Rate Offset Compensation - An Overlap-Save Based Approach

Joerg Schmalenstroeer, Reinhold Haeb-Umbach
Department of Communications Engineering, Paderborn University, Germany
{schmalen, haeb}@nt.uni-paderborn.de

***Abstract*—Distributed sensor data acquisition usually encompasses data sampling by the individual devices, where each of them has its own oscillator driving the local sampling process, resulting in slightly different sampling rates at the individual sensor nodes. Nevertheless, for certain downstream signal processing tasks it is important to compensate even for small sampling rate offsets. Aligning the sampling rates of oscillators which differ only by a few parts-per-million, is, however, challenging and quite different from traditional multirate signal processing tasks.**

In this paper we propose to transfer a precise but computationally demanding time domain approach, inspired by the Nyquist-Shannon sampling theorem, to an efficient frequency domain implementation. To this end a buffer control is employed which compensates for sampling offsets which are multiples of the sampling period, while a digital filter, realized by the well-known Overlap-Save method, handles the fractional part of the sampling phase offset. With experiments on artificially misaligned data we investigate the parametrization, the efficiency, and the induced distortions of the proposed resampling method. It is shown that a favorable compromise between residual distortion and computational complexity is achieved, compared to other sampling rate offset compensation techniques.

***Index Terms*—Overlap-Save method, sampling rate offset, resampling**

I. Introduction

Sensor networks promise a flexible infrastructure for multi-channel recording setups. Due to their distributed nature, the devices a sensor network is comprised of usually lack a common sampling clock. However, if each device has its own oscillator, there will be unavoidable deviations in the sampling frequencies, even if all devices sample at the same nominal rate. The reasons are manufacturing differences between the (crystal) clocks and environmental factors, such as the device temperature, which affect the oscillator frequencies.

An often-cited example for distributed signal acquisition is a Wireless Acoustic Sensor Networks (WASN), where each sensor node hosts a single or an array of microphones and where the nodes are connected via a wireless link. The spatial diversity achievable by such a distributed sensor network allows for superior signal extraction and multi-channel signal processing compared to a single spatially compact microphone array which is possibly located far away from the signals of interest [1]. Since each device has its own oscillator driving the A/D-converters the sampling rates at each node will be slightly different and the signal streams will diverge over time. This has a detrimental effect on various acoustic processing algorithms, e.g., on source localization [2], beamforming [3] or blind source separation [4], as described in the given references.

Several solutions have been proposed to estimate such sampling rate offsets. One option is to exchange time stamps between the devices from which the offset can be estimated [5], [6]. Alternatively the properties of the sampled acoustic data streams are analyzed, from which estimates of the sampling rate offsets can be obtained, e.g., by evaluating coherence functions [7], [8] or correlations in the time [9] or frequency domain [10].

In this contribution we are, however, not concerned with the estimation of the offsets but with their compensation, i.e., with adjusting the sampling rate of different devices to a common value. The conceptually simplest solution is to use special hardware components, such as tunable oscillators, to adjust the sampling rates to the desired values, as proposed in [6]. However, in most scenarios the given hardware is unalterable and does not include such a tunable device, which is why one has to resort to software solutions.

Traditional digital-to-digital sampling rate conversion methods fail on the task of changing sampling rates, which differ only by a few parts per million (ppm). They usually target rational sampling rate conversion rates $r = L/M$ where L and M are small integer numbers [11]. As the maximum of the two factors (L or M) determines the width of the anti-aliasing filter's passband, a resampling by a few ppm will easily create unrealizable filter constraints.

If signal processing speed and computational efficiency is the major objective, simple interpolation schemes can be used, e.g., linear, cubic or spline interpolation. However, these interpolators introduce frequency dependent distortions, which can have a detrimental effect on the subsequent signal processing tasks. A combination of upsampling and interpolation has been proposed in [4], where the signal is first interpolated by a factor of four and, subsequently, a fourth order Lagrange polynomial is employed to calculate the interpolated values. Here, the required low-pass filter in the first upsampling step and the Lagrange interpolation limit the achievable precision.

The optimal interpolation solution is known from the Nyquist-Shannon sampling theorem: The continuous-time signal is reconstructed from the discrete sequence of samples by means of a sinc interpolation, and is resampled with the desired sampling rate. However, the major drawback of this approach is its computational inefficiency. For each new sample the weighted sum of a fairly large number of sinc function values has to be calculated. Unfortunately, the arguments at which the values of the sinc function are required for the summation, is constantly changing, because a constant difference in sampling rate leads to a linearly increasing (or decreasing) sampling phase.

The popular Overlap-Save method (OSM) is a widely used approach for handling computationally demanding signal processing steps efficiently in the frequency domain. Its block-oriented processing is well prepared for handling both streaming data and off-line data. It has been used in [12] for resampling with rational factors (small L and M). In [13], the authors propose to use the OSM for multi-band mixing and downsampling. That approach, however, requires the Fast Fourier Transform (FFT) length to be an integer multiple of the downsampling factor, which is an unrealistic assumption in Sampling Rate Offset (SRO) compensation tasks.

In this paper we show how to combine the Overlap-Save method with the signal reconstruction according to the Nyquist-Shannon sampling theorem. Besides its suitability for processing streaming data, we will show that it is scalable in terms of precision and computational demands. Further we show that it compares favorably with other time and frequency domain interpolation techniques on artificially generated pseudo-noise data.

II. Ideal Signal Reconstruction

From the Nyquist-Shannon sampling theorem it is well-known how to perfectly reconstruct a bandlimited signal from its samples $x(n)$: Applying an ideal low-pass filter to the Fourier transform of the discrete time signal gives the Fourier transform $X(j\omega)$ of the continuous time signal:

$$X(j\omega) = \left[\sum_{n=-\infty}^{\infty} x(n)e^{-j\omega nT}\right] \cdot T\,\mathrm{rect}\left(\frac{\omega}{2W}\right), \quad (1)$$

where T is the time between two samples and W is the maximum occurring frequency in the signal $x(t)$. The continuous time signal $x(t)$ is then recovered by inverse Fourier transform

$$x(t) = T\frac{W}{\pi}\sum_{n=-\infty}^{\infty} x(n)\,\mathrm{sinc}\left(\frac{W}{\pi}(t-nT)\right), \quad (2)$$

with $\mathrm{sinc}(n) = \sin(\pi n)/(\pi n)$. If we choose the sampling time $T = \pi/W$, i.e., sampling with Nyquist rate, we can express the time domain signal by

$$x(t) = \sum_{n=-\infty}^{\infty} x(n)\,\mathrm{sinc}\left(\frac{t-nT}{T}\right). \quad (3)$$

This reconstructed signal can be sampled with any arbitrary rate full-filling the Nyquist-Shannon sampling theorem.

III. Resampling by Reconstruction

Let us assume we have two sensor nodes, R and S, sampling the same signal at slightly different sampling rates. We select the sampling rate $f_S = 1/T_S$ of node S to be the reference sampling rate and define the SRO ϵ between nodes S and R to be:

$$f_R = (1+\epsilon)\cdot f_S \Leftrightarrow \frac{T_S}{T_R} = (1+\epsilon). \quad (4)$$

Aligning the two sample sequences can be done as follows: First reconstruct the continuous time signal $x(t)$ from the discrete time sequence $x_R(n)$ via (3) and subsequently sample it at equidistant points $t = m\cdot T_S$ with the sampling frequency of node S. To be realizable, the summation in (3) has to be constrained to a finite number of values. Using a window of $(2\cdot L+1)$ values gives

$$\begin{aligned} x'_S(m) &= \sum_{n=\tilde{n}-L}^{\tilde{n}+L} x_R(n)\ \mathrm{sinc}\left(\frac{mT_S - nT_R}{T_R}\right) \quad (5)\\ &= \sum_{n=\tilde{n}-L}^{\tilde{n}+L} x_R(n)\ \mathrm{sinc}\left((1+\epsilon)m - n\right), \quad (6)\end{aligned}$$

where $x_R(\tilde{n})$ is the sample in the sequence of node R, which is temporally closest to mT_S. The parameter L determines the computational complexity and the achievable interpolation precision. As the sinc function decreases only with a factor of $1/n$ over time, fairly large values are required, e.g., $L > 64$, to keep the approximation error small. In the following we refer to this approach as “sinc interpolation”.

To compute one output sample the described sinc interpolation has to calculate $(2\cdot L+1)$ sinc function values, apply them as weights to the samples $x_R(n)$, and sum up the terms. Assuming a sampling frequency of $16\,\mathrm{kHz}$ and $L = 64$ this amounts to more than 4 million operations per second, just for resampling! To reduce the complexity and in parallel keep the precision high we investigate an Overlap-Save method implementation in the following.

IV. Overlap-Save Resampling

Our goal is to approximate the reconstruction of (6) by a linear convolution. This would enable an efficient implementation in the frequency domain by utilizing an OSM.

Eq. (6) can be written as

$$x'_S(m) = \sum_{n=\tilde{n}-L}^{\tilde{n}+L} x_R(n)\ \mathrm{sinc}\left((1+\epsilon)(m-n) + \epsilon\cdot n\right). \quad (7)$$

The index n in $(\epsilon\cdot n)$ can be approximated by the center value $\tilde{n}$ if the window size is small compared to the SRO changes within the window:

$$x'_S(m) \approx \sum_{n=\tilde{n}-L}^{\tilde{n}+L} \underbrace{x_R(n)}_{a_n}\ \underbrace{\mathrm{sinc}\left((1+\epsilon)(m-n) + \epsilon\cdot\tilde{n}\right)}_{b_{m-n}}, \quad (8)$$

which is a linear convolution between the signals a_n and b_n.

Eq. (8) can be realized efficiently in the frequency domain, where the input block size determines the degree of approximation. Since the block size B determines both the computational complexity and the error introduced by the approximation, it has to be selected carefully in accordance to the SRO ϵ. Larger SROs require smaller block sizes to keep the error small. Note, that the finite sum of (8) causes distortions of the spectrum of the sinc function, including the Gibbs-Phenomenon, which can be reduced by windowing the sinc function with a symmetric Hann-window.

The offset $\tilde{n}$ within the l-th block is given by

$$\tilde{n}(l) = (1+\epsilon)\cdot(lB + B/2), \tag{9}$$

i.e., a value linearly changing over time if the SRO is assumed to be constant. As the input buffer of the OSM has to hold the values closest to the sample to be interpolated, we split (9) in two parts: the first part counting the full (integer) samples

$$\tilde{n}_i(l) = \lfloor (1+\epsilon)\cdot(lB + B/2) \rceil, \tag{10}$$

and the second part containing the difference between $\tilde{n}(l)$ and its rounded value:

$$\tilde{n}_d(l) = \tilde{n}_i(l) - (1+\epsilon)\cdot(lB + B/2), \tag{11}$$

i.e., the fractional part. Here, $\lfloor\rceil$ denotes rounding towards the next integer value.

The OSM will take care of $\tilde{n}_d(l)$, while $\tilde{n}_i(l)$ will be treated by an input buffer control. Since $\tilde{n}_d(l)$ is always smaller than half a sample, the delay can be realized by a multiplication with an exponential function in the frequency domain, without getting severe cyclic wrap-around effects. Calculating the values of $b_n = \operatorname{sinc}((1+\epsilon)n + \epsilon\cdot\tilde{n})$ for $\tilde{n} = 0$ and applying the FFT transform in advance reduces the computational complexity of the Overlap-Save method significantly, since it saves one FFT operation per block.

The handling of the delay $\tilde{n}_i(l)$ is integrated in the input buffer of the Overlap-Save method, where the block shift is manipulated by $\tilde{n}_i(l)$. So, instead of shifting the sinc function we shift the input buffer. A possible implementation is depicted in Fig. 1.

This approach is called "OSM method" in the experiments.

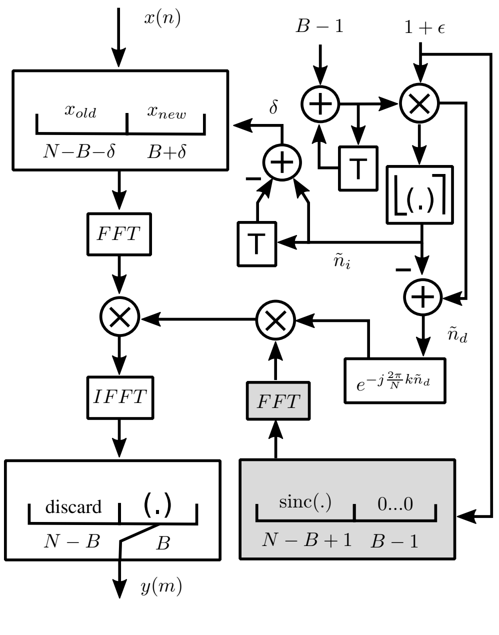

Fig. 1. OSM block diagram, grey blocks are used only if SRO changes

V. Frequency domain SRO compensation

In [10] the authors proposed to compensate for a SRO in the frequency domain by a multiplication with a complex phase. We shortly summarize the idea, explain how to extend it to data stream processing and use it for comparisons in the experiments.

A sampling rate offset ϵ introduces an increasing delay between the sample streams of nodes R and S. If the Short Time Fourier Transform (STFT) is applied on both data streams we can approximate the dependency between them following [14] by

$$X_R(l,k) \approx X_S(l,k)\cdot e^{-j\frac{2\pi}{N}(\frac{B}{2}+lB)k\epsilon}, \tag{12}$$

where N is the FFT size, B the block shift and $X_R(l,k)$ denotes the k-th bin of the l-th's block STFT result. Note, that zero padding is applied as the difference between FFT size N and block size B is filled up with zeros. This, again, reduces negative cyclic wrap around effects. Additionally, it is assumed that both streams started synchronously with sampling at $l = 0$ or had at least been roughly synchronized.

Extending the frequency domain approximation of (12) to a frequency domain resampling procedure can be done as follows. At first, an analysis window, i.e., a periodic hann window in our case, is applied to each input block. The block overlap is set to half of the block size B to avoid a synthesis window. The result of the N-point FFT is multiplied with the exponential

$$p(k) = e^{-j\frac{2\pi}{N}[\tilde{n}_d(l)\cdot\epsilon]k}. \tag{13}$$

The integer part of the delay is again compensated for by the input buffer using shift operations. Subsequently, the Inverse Fast Fourier Transform (IFFT) is applied and the result is added to the result of the previous block with an overlap of $B/2$. In the following, this method is denoted as "STFT method".

VI. Experiments

All experiments are conducted on artificially generated data. To this end a set of randomly weighted sinusoid functions are superposed to obtain bandlimited pseudo-random noise which can be sampled precisely at variable SROs for input and reference data. We created four sets of data: The wideband signal occupies the frequencies between 50 Hz and 7.95 kHz, while lower-band, middle-band and upper-band pseudo-noise signals are bandlimited to (50 Hz - 3 kHz), (3 kHz - 6 kHz) and (6 kHz - 7.95 kHz), respectively. For all experiments, the nominal sampling rate was 16 kHz.

The performance of the interpolation methods is measured in terms of Signal-to-Interference-Noise Ratio (SINR), which we define as follows:

$$\mathrm{SINR} = 10\log_{10}\left\{\frac{\sum\limits_m x_s^2(m)}{\sum\limits_m [x_s(m) - x_s'(m)]^2}\right\}, \tag{14}$$

where the sums include all samples m, excluding possible transient phenomena during initialization.

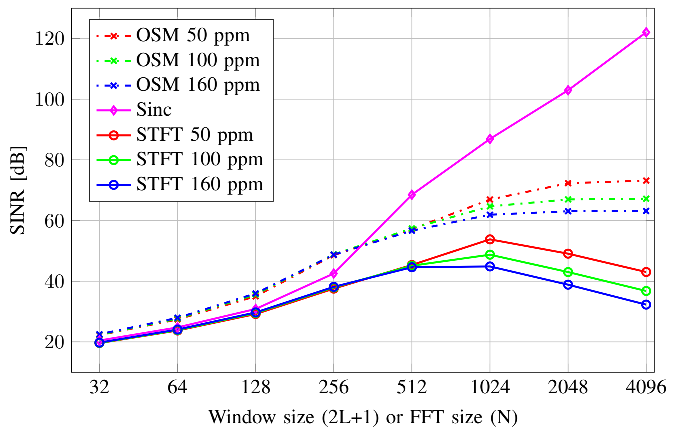

Fig. 2. Comparison of SINR values for resampling techniques on wideband noise and varying window (sinc) or FFT (OSM, STFT) sizes. OSM uses a block shift of $B = 8$.

At first, the STFT and the OSM approaches are compared against the sinc interpolation method in Fig. 2 on a logarithmically scaled abscissa stating either the window size $(2L+1)$ for the sinc interpolation or the FFT size (N) for the OSM and STFT methods. For small window and FFT sizes the sinc method performs somewhere between the STFT and the OSM. As the OSM models the same sinc function as the time domain sinc interpolation it is surprising that the two do not show the same performance. We attribute this to the *hann*-window applied to the sinc function in the sinc interpolation method. With increasing window sizes the influence of the *hann*-window vanishes and the sinc method becomes superior to all other approaches. Unfortunately, the computational complexity of the sinc interpolation method grows rapidly with the window size (see Fig. 5 for a comparison).

Both the OSM and STFT method profit from larger FFT sizes, up to a certain value. After $N = 1024$ the SINR achieved by the OSM method levels off, reaching plateaus depending on the SRO. The simpler STFT approach reaches a SRO-dependent maximum SINR between $N = 512$ and $N = 1024$, where two opposing trends meet. For small block sizes the analysis window attenuation limits the performance of the STFT interpolation. For larger block sizes the analysis window's influence on the STFT result is reduced, but the approximation of (12) is violated to an increasing degree . This harms the precision and limits the possibility to arbitrarily enlarge the FFT size. From the experiments the parameters $B = 512$ and $N = 1024$ seems to be a good trade-off.

Fig. 3 shows the SINR performance of some interpolation methods w.r.t the bandwidth of the input signal. For the SRO dependent approaches, i.e., Overlap-Save method and STFT, small (50 ppm) and medium (160 ppm) SROs are selected. As the parameterization of the Spline, Lagrange and sinc interpolation are independent of the SRO value we average their SINR values across all simulated SROs. We compare the Overlap-Save method also against the Lagrange interpolation approach mentioned in [4], where firstly the signal is interpolated by a factor of four and, subsequently, the interpolated signal is

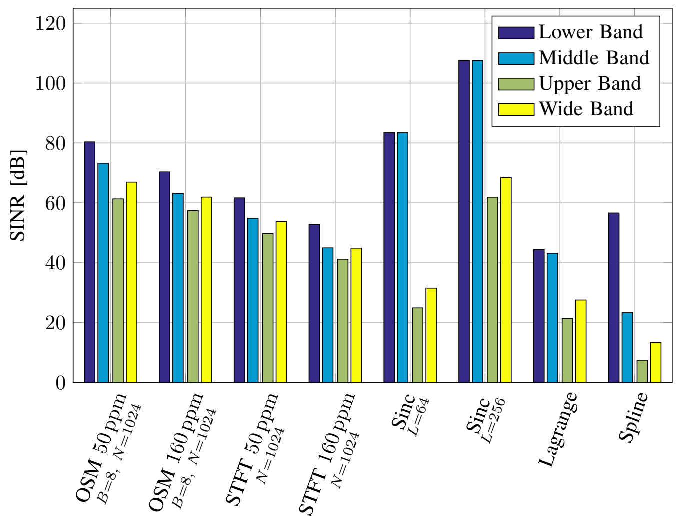

Fig. 3. Precision of different resampling methods with respect to input signal bandwidth.

decimated by a Lagrange method. Due to the fact that the interpolation step requires a low-pass filter, a Chebyshev-Type-I-Low-pass filter in our case, the precision of the result is limited as the filter modifies the signal during the upsampling step.

For lower band signals even simple and fast approaches show good SINR performance, e.g., Spline exceeds 55 dB in this subset. All approaches show higher errors for signals having higher frequency components. Lagrange handles lower and middle band signals satisfactorily, but performance drops rapidly for upper band signals, as the upsampling low pass filter limits the precision in this band.

Although the sinc interpolation ($L = 256$) outperforms the Overlap-Save method ($B = 8$, $N = 1024$) in terms of SINR values for lower and middle band signals, both approaches reach nearly the same performance for wideband signals in case of small SROs. However, the Overlap-Save method is a factor of 4 faster than the sinc interpolation (see Fig. 5).

A. Parameter Selection

Selecting the optimal parameters is an important part of the system design. For the proposed Overlap-Save method it includes selecting the block size and the FFT size. The block size determines the block delay, the computational complexity and the error due to the approximations. The advantage of larger block sizes, lowering the number of operations per second, is bought at the expense of smaller SINR values for larger SROs, because the approximation in (8) is more and more violated. The FFT size corresponds to the size of the sinc function, which is used to interpolate the new values. Here, a larger value promises better SINR values, but also requires more operations per block.

We propose to select the parameters for block and FFT size depending on the desired interpolation precision (SINR) and the SRO value. Figure 4 shows the necessary parameter values and the corresponding average processing time per 1 s length of input data for three targeted SINR values. The solid lines in the left plot show the block sizes and the dashed lines display the FFT sizes. For example, if a SINR larger than $60\,\mathrm{dB}$ for an SRO of $50\,\mathrm{ppm}$ is desired, a block size of $B = 32$ and a FFT size of $N = 1024$ is a suitable parameter set. The processing time would be around $0.08\,\mathrm{s}$ per $1\,\mathrm{s}$ of input data (compare right plot blue curve).

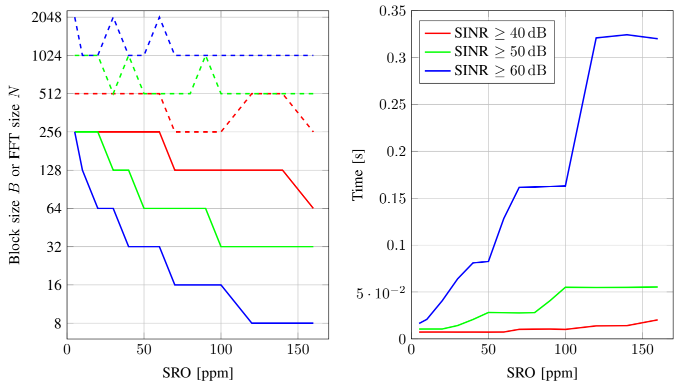

Fig. 4. OSM parameter selection for certain SINR limits. Left plot shows block size (solid line) and FFT size (dashed line) versus SRO. Right plot depicts the average processing time per $1\,\mathrm{s}$ audio segment for the selected parameters.

At last we present some experimental results concerning the computational complexity of the interpolation methods. Fig. 5 displays the average processing time versus the achieved SINR values for different parameterizations of the sinc, OSM and STFT methods. The STFT method shows a good performance in terms of computational complexity, but the precision remains limited. Higher SINR values are achievable with the OSM or the sinc method, where the OSM method offers a wide range for selecting an appropriate trade off between complexity and precision.

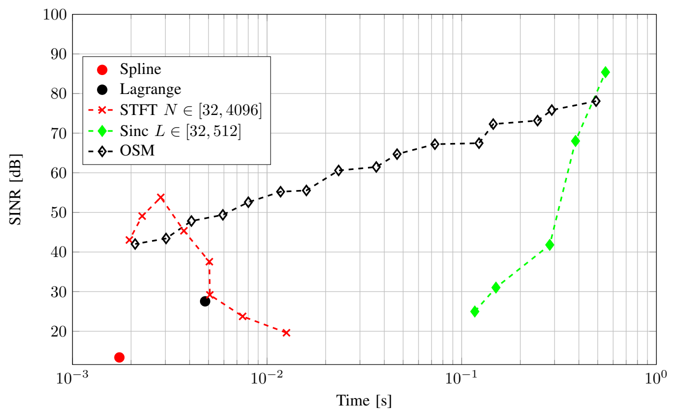

Fig. 5. Average processing times per $1\,\mathrm{s}$ audio segment for different interpolation methods (Intel Xeon CPU E3-1275 v5 @ $3.60\,\mathrm{GHz}$, SRO $50\,\mathrm{ppm}$).

VII. Summary

We presented a flexible approach for compensating sampling rate offsets in the frequency domain. To this end the computationally complex sinc time domain interpolation was approximated by a linear convolution. This enabled the efficient realization in the frequency domain by an Overlap-Save method. The choice of its parameters block shift and FFT size allow to trade off resampling precision in terms of the ratio between signal and interpolation noise with computational complexity.

Acknowledgment

This work was supported by *Deutsche Forschungsgemeinschaft* (DFG) under contract no. SCHM 3301/1-1 within the framework of the Research Unit FOR2457 “Acoustic Sensor Networks”.

References

[1] A. Bertrand, “Applications and trends in wireless acoustic sensor networks: A signal processing perspective,” in *2011 18th IEEE Symposium on Communications and Vehicular Technology in the Benelux (SCVT)*, Nov 2011, pp. 1–6.
[2] M. Pawig, G. Enzner, and P. Vary, “Adaptive Sampling Rate Correction for Acoustic Echo Control in Voice-Over-IP,” *IEEE Transactions on Signal Processing*, vol. 58, no. 1, pp. 189–199, 2010.
[3] S. Wehr, I. Kozintsev, R. Lienhart, and W. Kellermann, “Synchronization of acoustic sensors for distributed ad-hoc audio networks and its use for blind source separation,” *Proc. IEEE Sixth International Symposium on Multimedia Software Engineering*, pp. 18–25, 2004.
[4] S. Markovich-Golan, S. Gannot, and I. Cohen, “Blind sampling rate offset estimation and compensation in wireless acoustic sensor networks with application to beamforming,” *Proc. International Workshop on Acoustic Echo and Noise Control (IWAENC)*, pp. 4–6, 2012.
[5] Q. M. Chaudhari, “A Simple and Robust Clock Synchronization Scheme,” *Communications, IEEE Transactions on*, vol. 60, no. 2, pp. 328–332, 2012.
[6] J. Schmalenstroeer, P. Jebramcik, and R. Haeb-Umbach, “A combined hardware-software approach for acoustic sensor network synchronization,” *Signal Processing*, vol. 107, no. 0, pp. 171–184, 2015.
[7] M. H. Bahari, A. Bertrand, and M. Moonen, “Blind Sampling Rate Offset Estimation for Wireless Acoustic Sensor Networks Through Weighted Least-Squares Coherence Drift Estimation,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 3, pp. 674–686, 2017.
[8] J. Schmalenstroeer, J. Heymann, L. Drude, C. Boeddecker, and R. Haeb-Umbach, “Multi-stage coherence drift based sampling rate synchronization for acoustic beamforming,” in *19th International Workshop on Multimedia Signal Processing (MMSP)*. IEEE, 2017.
[9] D. Cherkassky and S. Gannot, “Blind Synchronization in Wireless Acoustic Sensor Networks,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 25, no. 3, pp. 651–661, 2017.
[10] L. Wang and S. Doclo, “Correlation maximization-based sampling rate offset estimation for distributed microphone arrays,” *IEEE/ACM Transactions on Speech and Language Processing*, vol. 24, no. 3, pp. 571–582, 2016.
[11] T. Ramstad, “Digital methods for conversion between arbitrary sampling frequencies,” *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 32, no. 3, pp. 577–591, Jun 1984.
[12] S. Muramatsu and H. Kiya, “Extended overlap-add and -save methods for multirate signal processing,” *IEEE Transactions on Signal Processing*, vol. 45, no. 9, pp. 2376–2380, Sep 1997.
[13] M. Borgerding, “Turning overlap-save into a multiband mixing, downsampling filter bank,” *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 158–161, March 2006.
[14] S. Miyabe, N. Ono, and S. Makino, “Blind compensation of interchannel sampling frequency mismatch with maximum likelihood estimation in STFT domain,” *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 674–678, 2013.