RBS-based Time Synchronization Approach with Autonomous Acoustic Sensors – a Simulative Proof of Concept

Alexander Tessmer
Institute of Computer Science
Osnabrück University
Osnabrück, Germany
tessmer@uos.de

Leonhard Brüggemann
Institute of Computer Science
Osnabrück University
Osnabrück, Germany
brueggemann@uos.de

Nils Aschenbruck
Institute of Computer Science
Osnabrück University
Osnabrück, Germany
aschenbruck@uos.de

***Abstract*—Low-cost Autonomous Recording Units (ARUs), like the AudioMoth, and advances in AI-based classifiers have enabled researchers and hobbyists to increase automation of the previously manual discipline of species monitoring. Often, the evaluation of acoustic monitoring still has to be done manually and one particular reoccurring challenge in the automated evaluation is the clock drift of the distributed ARUs. In this paper, we present a proof of concept for a time synchronization approach based on Reference Broadcast Synchronization (RBS) that uses random environmental sound events like bird calls to synchronize the clocks of receiving recording units. The application of RBS in the acoustic monitoring context is evaluated by simulation with the TAWNS framework. Subsequently, a problem in using RBS with sound events is identified and mitigated.**

***Index Terms*—AudioMoth, ARUs, clock drift, Internet of Sounds, RBS, TAWNS, time synchronization**

I. Introduction

The Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES) records a rapid decline in biodiversity [15]. Traditionally, species monitoring is often done manually in field surveys [25] and requires considerable effort. Therefore, the potential to improve species monitoring of new solutions is regarded as high. Acoustic sensing is an emerging topic to provide a solution for this challenge, e.g., [26], [28]. Advances in AI-based classifiers, e.g., [16], [17] and the availability of small, low-cost, power-efficient, and smart monitoring devices, like the AudioMoth [14], provide a good perspective for automated acoustic species monitoring. These Autonomous Recording Units (ARUs) are increasingly popular [28]. They can provide low-disturbance, large-scale monitoring of sound-producing animals, e.g., birds or mammals such as bats. In the following, we focus on bird species monitoring as it is a well-researched topic that provides a good indicator for environmental health and ecosystem changes [4], [10], [12].

We face multiple challenges when deploying multiple ARUs as an acoustic sensor network for bird species monitoring. The topic includes the complex tasks of identification, localization, and counting of birds. They consist of multiple research areas and ultimately remain unresolved [28]. There are different types of solutions for parts of this complex task, but many are time-based approaches. For example, localization can use energy-based methods, e.g., [19], that assume a known loss of energy in regard to the signal propagation. For bird species monitoring, the emitted energy is unknown and the signal itself difficult to separate. However, time-based solutions, e.g., [6], have been applied on bird acoustics, where time differences were extracted manually, e.g., [8], [28]. One challenge in using ARUs for an automated audio monitoring sensor network is their clock drift, which impairs even manually extracted time difference. Current solutions for the time synchronization of ARUs either require manual effort or communication modules that reduce battery life, e.g., GPS [33]. This impairs the feasibility of ARUs in bioacoustic monitoring. At the moment, one has to decide to live with either a reduced runtime or a reduced accuracy due to clock drifts.

In this paper, we propose to apply a time synchronization approach based on Reference Broadcast Synchronization (RBS) to provide passive time synchronization for the most basic configuration of an ARU without the use of communication modules by using the bird calls as reference events. The key contributions of this paper are threefold:

1) We present a solution approach for a passive time synchronization of ARUs by using only the recorded sound events.
2) A challenge for using sound events with RBS is identified and a first approach for mitigation is proposed.
3) Possible impacts on accuracy are estimated with a first implementation in an evaluation with a Terrestrial Acoustic and Wireless Network Simulation (TAWNS) framework simulation.

The remaining part of this paper is organized as follows: First, we review the use-case context of bird species monitoring and the several research areas that constitute the foundation for the time synchronization of ARUs in Sec. II. Then, Sec. III will present our approach and Sec. IV the test setup simulation environment. We analyze preceding requirements and the application of RBS using reference sound events in

Sec. V. Sec. VI identifies the problem of applying traditional RBS to reference sound events and proposes a solution that is evaluated. Finally, Sec. VII concludes this paper.

II. Related Work

A. Clock Drift

Typically, the clock in ARUs consists of a counter, e.g., a hardware register, which counts a periodic signal from a crystal oscillator. Clock drift happens, if the time between actual signals differs from the expected time interval of the oscillator. This oscillation varies between crystals and is impacted by several environmental factors, also with varying sensitivity. Therefore, the clocks drift continuously with changing drift rates due to day and night temperature, pressure, and humidity changes, the level of excitation through the declining supplied battery power, and many more [34]. Subsequently, the clock offset to the actual time is comprised of three independent factors [33]: (1) The initial clock offset, which often necessitates a time synchronization at system setup. (2) An independent noise with deterministic and random components, e.g., delay of time stamping. (3) The clock skew due to the variation in oscillation times, which in a simple way could be modeled by a constant drift or a randomly changing drift, but is largely impacted by the environmental factors. This also necessitates regular time synchronizations to keep the clocks accurate.

B. Time Synchronization

The typical time synchronization method in the context of the internet or Wireless Sensor Networks (WSNs) is the Network Time Protocol (NTP) [20]. It synchronizes two nodes with a message round trip and taking two timestamps each for the sender and receiver. This benefits from having similar delay between the two send directions. It is a quick and simple method to achieve millisecond accuracy to a global time by hierarchically synchronizing two nodes. This is even more accurate, when only used locally in a single hop transmission. Here, its variation can be categorized into three parts: (1) sender delay, (2) propagation delay, and (3) receiver delay.

In a WSN context, if higher accuracy is needed or a global reference can not be obtained accurately enough with NTP, using Global Navigation Satellite Systems (GNSSs) for time synchronization is a common alternative or addition. It provides very accurate time synchronization with microsecond accuracy on at least a regional scale, depending on the particular GNSS used, like GPS [33]. Because of its popularity and ease of use, ARUs with GNSS modules or at least an attachment are prevalent. However, the use of GNSS modules increases the power consumption. Thus, it requires more investment in power infrastructure or manual maintenance. For ARUs this is a considerable factor.

For the specific acoustic type of WSN, the Wireless Acoustic Sensor Network (WASN), acoustic synchronization is an additional alternative that can be independent of the wireless communication between nodes [28]. In this case, the alignment between recorded acoustic signals is estimated by synchronizing sources within the signal in the presence of noise. We will differentiate this further in Sec. II-E. Since the signal source alignment is not only impacted by clock drift, but also by sound propagation, either the spatial distribution of microphones needs to be small, e.g. gathering all sensors at a single location for synchronization, or the location of the sound event has to be known to calculate the propagation delays. This method can alternatively be used just to align the acoustic signals, e.g., to improve the sound separation, but the accuracy of the time offset information is impacted by the clock drift, if used for further evaluation. We use a simple setup for the time offset estimation as presented in Sec. V-B and focus on using multiple time offsets for a more robust time synchronization based on RBS.

C. Reference Broadcast Synchronization (RBS)

Reference Broadcast Synchronization (RBS) [9] is an alternative time synchronization method to achieve much higher accuracy than NTP in the WSN context. A sensor node broadcasts a signal and only the time of the signal receivers is synchronized. Therefore, a minimum number of three nodes or two nodes and a separate reference event emitter is necessary. With randomized network nodes as reference signal emitters, the whole WSN can be locally synchronized. External timescales can also be included by synchronizing at least one node to this timescale, e.g., one node with GPS or NTP time synchronization. In comparison to NTP, RBS does not set the node clocks, but generates a clock conversion library for each node pair from the difference in reference event receive times. It assumes minimal variance in transmission time, which is true for radio signals, but much worse with sound propagation. The random noise in signal receive times is mitigated by averaging multiple reference events. To also include clock skew in the time synchronization, the average is upgraded to a least-squares linear regression. With RBS, if a subset of the nodes can be synchronized and there are sufficiently many reference events for overlapping subsets, the whole set of nodes can be synchronized. Increased spacing between nodes creates more subsets and the synchronization accuracy decreases proportional to the square root of the number of synchronization hops. In contrast, with larger sets the chance of one node being poorly synchronized is increased.

D. Terrestrial Acoustic and Wireless Network Simulation (TAWNS)

To the best of our knowledge, there are only few acoustic network simulators addressing bioacoustic use cases such as the one addressed in this paper. Most focus on indoor environments, such as pyroomacoustics [30], real-time audio reproductions [11], or simulating sounds in computer games [21]. For this work, we require a simulator capable to simulate more than spatial sound propagation in a 3D environment. We also simulate sensor-dependent features like the clock drift. To the best of our knowledge, there is only the TAWNS framework [2] that combines the two aspects of (wireless) sensor network simulation and bioacoustic sound simulation. This is enabled due to its modular design. For network

simulation, it uses the popular OMNeT++ framework [23], in which the well-known INET framework [22] is implemented. While mainly a framework capable of simulating different types of sensor networks, including various topologies and wireless communications, it also supports common time drift models for the sensor clocks TAWNS inherits all INET functionalities and more, combining them with the simulation of sound propagation. For that, it extends Scaper [29], a library for soundscape augmentation and generation, that by default does not support simulation of sound propagation. A custom-designed sound propagation model has been added in TAWNS. Additionally a list of model parameters has been derived from real measurements of bird acoustics. Consequently, TAWNS is extremely well suited for this work, in which we focus on correcting time drift in ARU recordings of bioacoustic signals.

E. Blind Source Separation (BSS)

In the context of separately recorded audio files the synchronization of audio files is often done manually and/or by emitting an artificial sound event that is much louder than the environment and therefore distinctive, e.g., a movie clapboard or gunshot [26], [28]. Extracting individual bird calls within the recorded sound presents a challenge. Background noise or other birds often interfere and present a problem. BSS addresses the extraction of the individual sound event from a composited signal without any prior information about it [5], [24]. This is a complex problem with multiple different approaches [7], [31]. Fortunately for the time synchronization, only a single feature of a sound event needs to be the reference between multiple recordings for a time difference estimation.

One of the most popular methods of time difference estimation is Generalized Cross-Correlation (GCC), which was originally introduced in 1976 [18]. The Fourier transform of two signals is calculated and its cross-spectral density is determined. This is finished by a weighting function like phase transform (PHAT) as a popular choice due to its robustness [18]. As a result, maximum peak shifts correlations determine the time shift between the signals. A popular alternative that focuses more on voice signal processing is calculating the cross-correlation of Mel-Frequency Cepstral Coefficients (MFCCs) [13]. Calculating the MFCCs also starts with the Fourier transform. Then, a Mel band-pass filter maps the spectrum onto the Mel scale by taking the logarithms of the powers at each of the Mel frequencies. Finally, the Discrete Cosine Transform (DCT) is calculated [1]. In our work, we utilize the cross-correlation of MFCCs to estimate time differences as it is robust to noise, an implementation is easily available, and it provides a solution to a challenge that is not in the focus of this paper.

F. Deployment Context

Time difference estimation methods use the time differences to infer information about the sound event. This is impacted by time differences that are independent of the sound event itself, like clock drift. In long-term deployments of ARUs this can accumulate into a large impact on the inferred results, since manual time synchronization takes effort and automatic time synchronization requires communication and adding a communication module increases effort in maintenance due to the larger power consumption. It should be noted that the AudioMoths can be synchronized with specific audio commands, which are embedded in played audio. However, its functionality, especially over large distances is only marginal. Also, the AudioMoths are not equipped with speakers by default. This also requires more effort or more complex setups. A passive time synchronization promises to provide the lowest effort solution. To the best of our knowledge, providing passive regular time synchronization for ARUs in the context of bird species monitoring has not been done. Our approach for time synchronization in this context should ultimately enable more accurate solutions for the complex task of bird species monitoring with a low entry point for hobbyists and researchers.

III. Time Synchronization Approach

We want to increase the accuracy of time-based sensing algorithms by providing time synchronization to the baseline ARU deployment. As the setting, we have the bird calls, i.e., randomly distributed audio source nodes with unknown location, and ARUs, i.e., audio recording nodes with known location and drifting clocks that lead to inaccurate recording timelines. For accurate time difference estimations, we need to align the timelines of the different recordings, i.e., we need timeline mapping functions between the nodes. We also need a time synchronization algorithm that can be applied subsequent to the monitoring task as we only gain access to the recording, when the data is manually collected.

The known time synchronization method for similar restrictions in the WSN radio communication context is RBS as it creates a clock conversion library for each node pair. In RBS, only the receive timestamp of the reference broadcast from a randomly chosen node is needed for the receiving nodes to coordinate the time synchronization. As the broadcast node does not have to be part of the system, we consider each bird call as a broadcast node message, which is called the reference broadcast event. Determining the broadcast origin in RBS is handled by the communication module and the position is not important as the propagation delay is considered negligible. Therefore, we need to extract timestamps from the recording that correspond to the same source. This BSS part can be modularly exchanged as long as it generates a time offset or timestamp of the received broadcast event from the same audio source. The RBS algorithm leverages randomly distributed radio packet arrival time offsets to increase accurate results with multiple broadcast events. We propose to use the varying sound receive times due to the random sound source locations to achieve a similar effect. We will consider the problems arising from this assumption in Sec. VI. Therefore, we initially follow the standard RBS algorithm to calculate the clock conversion library in our simulation.

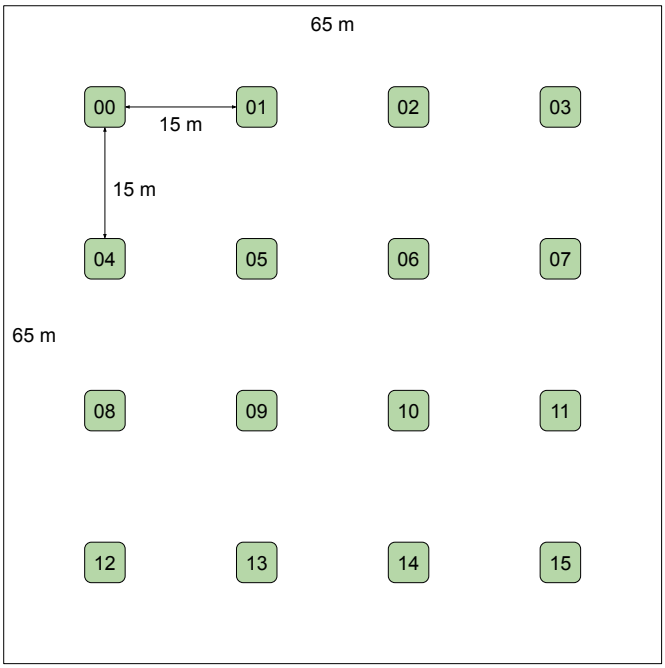

Fig. 1: Node placement for the test setup

IV. System Model

Our focus is on the context of acoustic monitoring with ARUs on the example of bird species monitoring. We consider the AudioMoths [14] as the used ARUs in their minimal hardware configuration as they are commonly used in this context. Each node is only equipped with a drifting Real-Time Clock (RTC), batteries, and the audio recording equipment. Therefore, no radio communication is possible. In the bird species monitoring context, the ARUs are typically deployed inside a forest with somewhat regular distances in an irregular shape, depending on the environment. For our proof of concept, we simplify this with a square grid based deployment, which is displayed in Fig. 1. As we identify in Sec. V, the placement has an impact on the time synchronization, but our proposed mitigation in Sec. VI works regardless of the specific deployment positions.

We use the TAWNS framework [2] for our simulative evaluation. The simulation framework enables the use of real bird call samples as foundation and generates noise, delay, and more according to the simulation setup. We selected the common redstart (Phoenicurus phoenicurus) with a distinctive bird call and a low level of forest noise as BSS is not the focus of this paper and this constitutes an easier scenario for the BSS task. Additionally, it should be noted, that a more complex BSS can also solve the task of sound event recognition. This identifies the nodes that were able to record the sound event and selects them as nodes for the time synchronization. Since the time synchronization of overlapping node subsets combines to a time synchronization for the whole set, we provide the proof of concept for a whole set, by evaluating only a subset. Therefore, the subset can be considered as the whole set and in our setup, all nodes are able to record the bird call so that sound event recognition is for all nodes specified as true.

Tab. 1: Test setup parameters

Parameter	Value
Simulation Duration	93 s
Simulation Area	65 m x 65 m
Recording Nodes	16
Node Distance	15 m
Border Distance	10 m
Sound events	30

The parameters of the simulation test setup are recorded in Tab. 1. For comparable results to RBS [9] with traditional radio broadcasts, we choose a setup of 16 recording nodes and 30 reference sound events. For distinct reference event separation, we define a window of three seconds for each sound event and a single window as simulation start and control sample, which results in a simulation time of 93 seconds. The four by four grid of the 16 recording nodes is spaced according to previous field experience, where all nodes are be able to record the sound event. In practice, a four times more sparse deployment should still include sufficient nodes for a time synchronization. We specify a simulation area of 65 by 65 meters and a spacing of 15 meters between the nodes, which leaves a 10 meter border.

V. Proof of Concept

For the evaluation of the proof of concept for our time synchronization approach, we focus on analyzing a single time synchronization for our set of nodes. We consider a small time window in the morning with a low forest noise background as the application context that is simulated with TAWNS. This could exemplify a time synchronization scheme, where the nodes are synchronized daily in the morning. All 16 ARU nodes are recording audio files with their RTCs as timeline and the raw recording is evaluated. For the evaluation of our RBS-based time synchronization approach, we have to consider several prerequisite aspects: The generation of the recording data is completed by TAWNS according to our system model. Clock drifts are simulated separately in TAWNS and can be combined afterwards. Lastly, we have to extract a timestamp or an offset between sound events from the recording data with BSS.

A. Clock Drift

There are two main challenges in simulating clock drifts: The model choice and the scaling. Both should preferably be solved by doing a hardware analysis. This will be future work due to lack in sufficient amount of hardware. We tested basic clock drift models and Fig. 2 shows a simulation of a random clock drift model with randomly changing clock skew for the 16 ARU nodes. The scaling of the clock drift is dependent on model parameters and the simulation time. Rhinehart et al. [28] specify ARU clock drifts of up to 10 seconds per day. This could be used as foundation for the simulation, but also

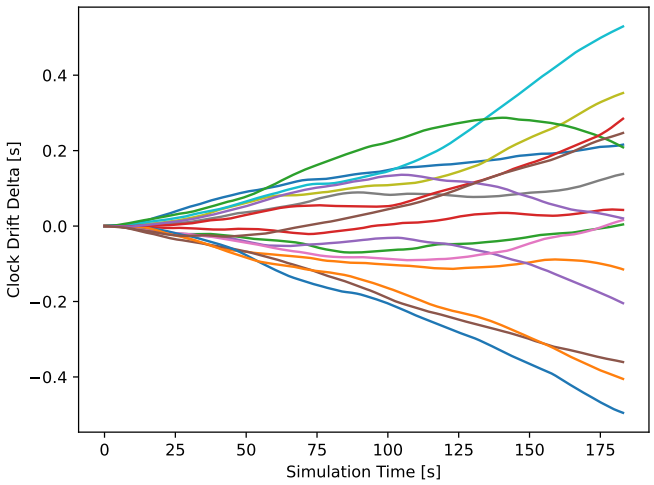

Fig. 2: Clock drift simulation for 16 nodes with the random clock drift model

introduces other variables like time synchronization frequency and initial clock offset due to the impact of the simulation time. For the application of bird species monitoring, this also includes restrictions due to the sound events not being created as part of the process, e.g., audio playback by a speaker node, but having to rely on existing environmental sound events. In conclusion, the simulation of clock drifts introduces too much test complexity for a proof of concept. Fortunately, with the random distribution of relative clock drifts around zero, we know the general time synchronization characteristics of our approach from RBS [9]. The missing difference to our approach is the impact from sound as the reference event medium, which is only dependent on the recording data. Also with the simple BSS setting as presented in the following section, there is no error in extracting the timestamp. So the impact of a simple clock drift model offset is immediately negated in the time synchronization and more complex models require further research. Therefore, we exclude the clock drifts in our analysis and only consider the raw audio files for the time synchronization.

B. Blind Source Separation (BSS)

As we only consider a simplified application scenario regarding the BSS, the requirement for BSS of extracting a timestamp or sound event offset from the recording data is not a challenge for our proof of concept. For verification, we manually analyzed the recording data to determine the correct sound event times to facilitate a ground truth. We compare a simple peak finding solution and a more robust algorithm from the audio-offset-finder Python library [27] that uses cross-correlation of standardized Mel-Frequency Cepstral Coefficients. The Mel-Frequency Cepstral Coefficients are more designed for speech recognition, but will suffice in this scenario and cross-correlation is already widely used in this context. Both BSS methods always successfully identify the sound event time or offset. Therefore, all following evaluations of time synchronizations are implemented with the cross-correlation algorithm. A more complex analysis of BSS for bird calls is not focus of this proof of concept and should be done with much more complex bird call and noise test setups in separate work. For the time synchronization, we only need the event times, which we were able to extract from the simplified setup for our proof of concept.

C. RBS with Reference Sound Events

After the successful event time extraction by BSS, each reference sound event generates a set of offsets for each pair of recording nodes, which can be used to create the RBS clock conversion library. Since our setting does not include clock skew, we can consider the average of the sound event offsets instead of the least-squares linear regression. We define the average of all node clocks as the reference absolute time, which simplifies the automated coordination of a local timescale. Further, with no clock drift in this proof of concept evaluation this also corresponds to the global simulation time.

The resulting calculated node offsets are displayed in Fig. 3. A first expectation for the results is a decreasing offset with increasing number of reference events. This would be the expected result of RBS. It is also the case here and would probably get more prominent with more replications. However, the offsets do not converge into a single grouping, but rather into three. The four center nodes and the four corner nodes create their own separate offset grouping. This originates from the slow sound propagation in comparison to radio and the concept that the average distance of the node positions to other random points in the simulation area is not the same. The group spacing is impacted by node distance and the formation of distinct groups by the symmetrical simulation setup. In conclusion, due to the much larger variations in sound event receive times, this impact can not be ignored for reference sound events as RBS assumes with radio signals. However, for example, location determination of bird calls can also work with four nodes, which also largely mitigates this problem.

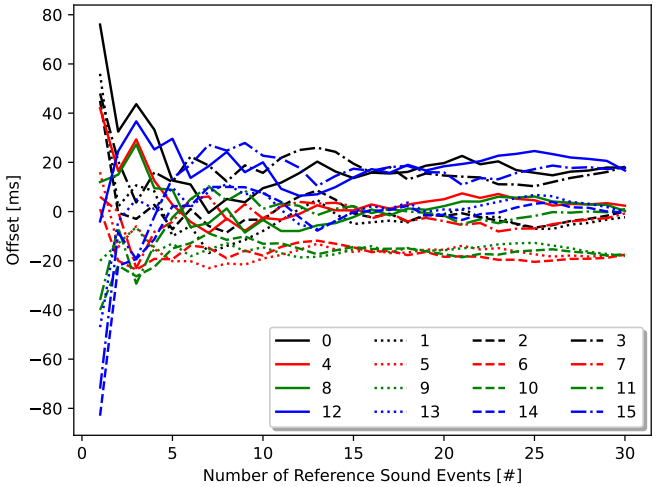

Fig. 3: Offset analysis of basic RBS with sound events (legend matches node placement, color for row, line style for column)

Two nodes always have the same distance to their average and more nodes can be arranged in that regard corresponding to the area, e.g., four nodes in square shape in a square.

VI. Improvements

In the previous, we have a decrease in accuracy, due to the greatly varying sound event receive times Our proposed approach to solve this, is to take the known node positions into account for the time synchronization. For the proof of concept, we solve the mathematical *average point to random points distance in a square* problem that is derived from the *average distance between two points* problem [3] to mitigate the impact in receive variation. In our simulation setup, the nodes are symmetrically placed in three circles around the simulation area center. Each circle corresponding to different values for the mathematical *average point to random points distance in a square* problem. We propose to solve this problem for each node position, which makes the solution independent of node placement. Subsequently, the average of all nodes that take part in a specific time synchronization event is taken and subtracted from each node. This determines the offset of each node in terms of distance that has to be corrected for. The distance offset can be converted into the time offset by introducing the speed of sound propagation. It should be noted that the speed of sound is dependent on several environmental factors and introduces a new source of errors. However, the impact of this error should be very small in comparison to the impact of sound event distribution or sound event detection range. The approach works well in a defined simulation area and supports the proof of concept. For a time synchronization of a subset of nodes or in an area with open borders, this solution has to be improved in future work. Modeling sound event recognition distances could solve this issue. Alternatively, the use of bird call location algorithm could provide the exact location of sound events and enable an offset shift for each event and not only after several iterations.

For our proposed offset shift based on average distances, we start the calculations with the formula for the euclidean distance between two points

$$f_p(x,y) = \sqrt{(x-p_x)^2 + (y-p_y)^2}, \tag{1}$$

where $f_p(x,y)$ is the distance between the points and p_x and p_y are the respective x- and y-coordinates of a point, we define as our fixed point p, while x and y are the respective x- and y-coordinates of the other point, we want to calculate the distance to. For the mathematical *average point to random points distance in a square* problem, we have to solve the formula

$$d(n_i) = \frac{1}{A} \iint_S f_p(x,y)dxdy, \tag{2}$$

where $d(n_i)$ is the average distance from the point of node i to all points in the simulation area square, A is the simulation area size, and S is the simulation area shape, i.e., the square. This formula can also be applied to different simulation area shapes, which simplifies the application of our proposed solution in real use cases. However, sound event recognition distances still have to be considered for a real use application. In practice, we can also calculate the average over multiple points, which can be iterated over the whole area or randomly chosen. For our square shape, we calculate the simplified formula

$$d(n_i) = \frac{1}{m_x * m_y} \lim_{\delta x,y \to 0} \sum_{y=0}^{m_y} \sum_{x=0}^{m_x} f_p(x,y), \tag{3}$$

where m_x and m_y are the respective x- and y-lengths of the simulation area square. Subsequently, we need the average of all $d(n_i)$ as offset reference, which we calculate with the formula

$$d_{avg}(N) = \frac{1}{|N|} \sum_{n_j \in N} d(n_i), \tag{4}$$

where $d_{avg}(N)$ is the offset reference and N is the set of nodes, which can be attributed to a specific synchronization event. Finally, to determine the additional offset for each node in this specific synchronization event, we calculate the formula

$$t(n_i) = \frac{d_{avg}(N) - d(n_i)}{c}, \tag{5}$$

where $t(n_i)$ is the inverse offset that has to be added to node i for the offset correction and c is the speed of sound.

With the shift of each time synchronization node offset by the calculated additional offset, we get the results presented in Fig. 4. As displayed, the separated offset groupings disappear, which makes this a successful solution. Therefore, we provide the proof of concept and the RBS-based and adapted time synchronization method to facilitate the synchronization of ARU audio data for bird species monitoring is feasible. Currently, there are no established automated bird species monitoring solutions, so the result evaluation depends on the future application. To compare automated bird classification results, it is only necessary to negate the constantly worsening

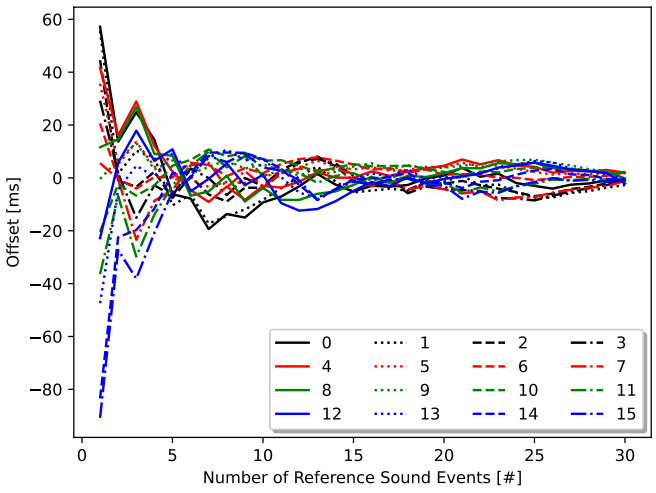

Fig. 4: Offset analysis of our adapted time synchronization approach (legend matches node placement, color for row, line style for column)

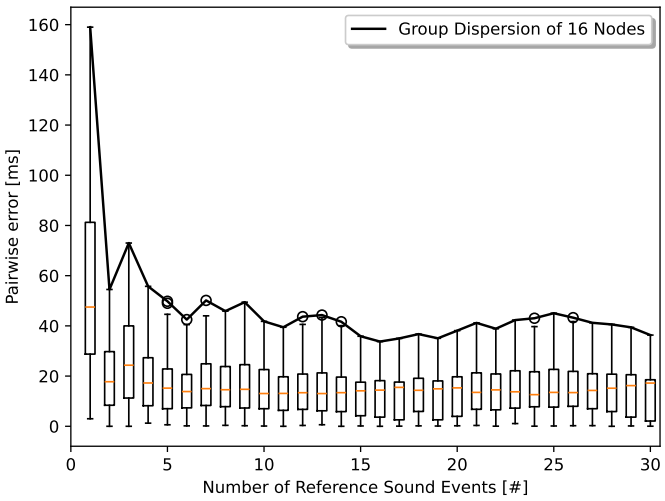

Fig. 5: Group dispersion (i.e., maximum pairwise error) analysis with 16 nodes (pairwise error boxes)

clock offset and an accuracy of approximately a second is acceptable. This requirement is more than satisfied. Although for a location determination, the results that roughly compare to an additional location error of up to 10 meters are fairly inaccurate. Especially, if the node distances of 15 meters are considered. However, in a practical deployment, there would be less nodes placed within such a deployment area. This large amount is only meant for the time synchronization evaluation. Furthermore, the bird location in current manual bird species surveys is often only recorded as an approximation on a map, if at all. Therefore, a comparable automated survey would be possible with the achieved synchronization results. Even so, for the aim of accurate bird localization the time synchronization results have to be improved in the future.

For a comparison with traditional RBS [9], we also determine the group dispersion (i.e., maximum pairwise error) in our test simulation. This is displayed in Fig. 5. It can be derived that initially the pairwise error is still high due to the highly variable center point of the low number of reference sound events, but decreases quickly with only very few reference sound events. The low pairwise error after only two sound events is an artifact of a luckily mirrored reference sound event and should disappear with more repetitions that were not possible due to time constraints and will be done in future work. It should also be noted, that group dispersion increases with the number of nodes synchronized, especially with added clock drifts, as can be derived from the box plot of the pairwise errors. However, with a full deployment reducing the number of synchronization nodes increases the synchronization hop count between the node subsets and decreases accuracy in that regard. The impact of this relation will be evaluated in future work after a more in-depth evaluation in a realistic deployment of this new proposed RBS-based and adapted time synchronization method.

VII. Conclusion

This paper presents a time synchronization method of ARU audio recording data for the example of bird species monitoring. A simplified simulation setup for a proof of concept with the TAWNS framework is specified. We discuss the complexity due to environmental variables, clock drift, and BSS. Furthermore, we especially analyze impact and simplifications for the aspects of clock drift and BSS. We show that the concept of RBS can be applied to audio data with reference sound events, but is impaired by the increased variation in propagation time in comparison to radio transmission. Subsequently, we identify the specific problem of separate synchronization convergences of node groups and propose possible improvements by using the node locations to estimate the convergence offsets. Finally, we specify and implement a solution to counter this impact in our simulation and discuss the successful results as well as shortcomings for general application, which mainly originate from the well-defined test area and sound event detection ranges that are more open and complex in real world settings.

In future work, we would like to do a proper hardware analysis of clock drifts to facilitate an evaluation with realistically drifting clocks. Additionally, further improvements to the time synchronization method need to be made to facilitate implementation in real hardware. This then should be evaluated in a real deployment. The ARUs without communication modules are currently prevalent for many long-term, minimum maintenance deployments, but ultimately the recordings are collected and contribute to the monitoring and understanding of our environment through the audio modality as is one aim of the Internet of Sounds (IoS) research agenda [32]. The proposed approach may lead to an improvement of the accuracy of acoustic monitoring with ARUs in the future.

References

[1] Z. K. Abdul and A. K. Al-Talabani, "Mel frequency cepstral coefficient and its applications: A review," *IEEE Access*, vol. 10, pp. 122 136–122 158, 2022.

[2] L. Brueggemann, B. Schuetz, and N. Aschenbruck, "TAWNS - A Terrestrial Acoustic and Wireless Network Simulation Framework," in *EAI 15th International Conference on Performance Evaluation Methodologies and Tools*, 2022.

[3] B. Burgstaller and F. Pillichshammer, "The average distance between two points," *Bulletin of the Australian Mathematical Society*, vol. 80, no. 3, p. 353–359, 2009.

[4] G. E. Canterbury, T. E. Martin, D. R. Petit, L. J. Petit, and D. F. Bradford, "Bird Communities and Habitat as Ecological Indicators of Forest Condition in Regional Monitoring," *Conservation Biology*, vol. 14, no. 2, pp. 544–558, 2000.

[5] J.-F. Cardoso and B. Laheld, "Equivariant adaptive source separation," *IEEE Transactions on Signal Processing*, vol. 44, no. 12, pp. 3017–3030, Dec. 1996.

[6] M. Cobos, F. Antonacci, A. Alexandridis, A. Mouchtaris, and B. Lee, "A Survey of Sound Source Localization Methods in Wireless Acoustic Sensor Networks," *Wireless Communications and Mobile Computing*, vol. 2017, no. 1, p. 3956282, 2017.

[7] Y. Dai, J. Yang, Y. Dong, H. Zou, M. Hu, and B. Wang, "Blind source separation-based IVA-Xception model for bird sound recognition in complex acoustic environments," *Electronics Letters*, vol. 57, no. 11, pp. 454–456, 2021.

[8] J. B. David R. Wilson, Matthew Battiston and D. J. Mennill, "Sound finder: a new software approach for localizing animals recorded with a microphone array," *Bioacoustics*, vol. 23, no. 2, pp. 99–112, 2014. [Online]. Available: https://doi.org/10.1080/09524622.2013.827588

[9] J. Elson, L. Girod, and D. Estrin, "Fine-grained network time synchronization using reference broadcasts," *ACM SIGOPS Operating Systems Review*, vol. 36, 06 2002.

[10] S. Fraixedas, A. Lindén, M. Piha, M. Cabeza, R. Gregory, and A. Lehikoinen, "A state-of-the-art review on birds as indicators of biodiversity: Advances, challenges, and future directions," *Ecological Indicators*, vol. 118, p. 106728, Nov. 2020.

[11] M. Geier, J. Ahrens, and S. Spors, "The soundscape renderer: A unified spatial audio reproduction framework for arbitrary rendering methods," *Proceedings of the Audio Engineering Society—124th Audio Engineering Society Convention*, pp. 179–184, 05 2008.

[12] R. D. Gregory and A. van Strien, "Wild Bird Indicators: Using Composite Population Trends of Birds as Measures of Environmental Health," *Ornithological Science*, vol. 9, no. 1, pp. 3–22, Jun. 2010.

[13] R. Gupte, S. Hawa, and R. Sonkusare, "Speech recognition using cross correlation and feature analysis using mel-frequency cepstral coefficients and pitch," in *2020 IEEE International Conference for Innovation in Technology (INOCON)*, 2020, pp. 1–5.

[14] A. P. Hill, P. Prince, J. L. Snaddon, C. P. Doncaster, and A. Rogers, "AudioMoth: A low-cost acoustic device for monitoring biodiversity and the environment," *HardwareX*, 2019.

[15] Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services (IPBES), "Summary for policymakers of the global assessment report on biodiversity and ecosystem services," 2019.

[16] S. Kahl, A. Navine, T. Denton, H. Klinck, P. Hart, H. Glotin, H. Goëau, W.-P. Vellinga, R. Planqué, and A. Joly, "Overview of BirdCLEF 2022: Endangered bird species recognition in soundscape recordings," *CEUR Workshop Proceedings*, 2022.

[17] S. Kahl, C. M. Wood, M. Eibl, and H. Klinck, "BirdNET: A deep learning solution for avian diversity monitoring," *Ecological Informatics*, 2021.

[18] C. Knapp and G. Carter, "The generalized correlation method for estimation of time delay," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 24, no. 4, pp. 320–327, Aug. 1976.

[19] W. Meng and W. Xiao, "Energy-Based Acoustic Source Localization Methods: A Survey," *Sensors*, vol. 17, no. 2, p. 376, Feb. 2017.

[20] D. L. Mills, "Rfc1129: Internet time synchronization: The network time protocol," USA, 1989.

[21] J. Müller, "Audaspace," 2015, retrieved July 15, 2024 from https://audaspace.github.io/.

[22] OpenSim Ltd., "INET Framework," 2023, retrieved July 15, 2024 from https://inet.omnetpp.org/.

[23] ——, "OMNeT++ - Discrete Event Simulator," 2024, retrieved July 15, 2024 from https://omnetpp.org/.

[24] M. Pal, R. Roy, J. Basu, and M. S. Bepari, "Blind source separation: A review and analysis," in *2013 International Conference Oriental COCOSDA Held Jointly with 2013 Conference on Asian Spoken Language Research and Evaluation (O-COCOSDA/CASLRE)*, Nov. 2013, pp. 1–5.

[25] C. Pérez-Granados and J. Traba, "Estimating bird density using passive acoustic monitoring: A review of methods and suggestions for further research," *Ibis*, 2021.

[26] E. Piña-Covarrubias, A. P. Hill, P. Prince, J. L. Snaddon, A. Rogers, and C. P. Doncaster, "Optimization of sensor deployment for acoustic detection and localization in terrestrial environments," *Remote Sensing in Ecology and Conservation*, vol. 5, no. 2, pp. 180–192, 2019. [Online]. Available: https://zslpublications.onlinelibrary.wiley.com/doi/abs/10.1002/rse2.97

[27] Y. Raimond, S. Jolly, and C. Needham, "audio-offset-finder," 2024, retrieved July 15, 2024 from https://github.com/bbc/audio-offset-finder.

[28] T. A. Rhinehart, L. M. Chronister, T. Devlin, and J. Kitzes, "Acoustic localization of terrestrial wildlife: Current practices and future opportunities," *Ecology and Evolution*, vol. 10, no. 13, pp. 6794–6818, 2020.

[29] J. Salamon, D. MacConnell, M. Cartwright, P. Li, and J. P. Bello, "Scaper: A library for soundscape synthesis and augmentation," in *2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, 2017, pp. 344–348.

[30] R. Scheibler, E. Bezzam, and I. Dokmanić, "Pyroomacoustics: A python package for audio room simulation and array processing algorithms," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2018, pp. 351–355.

[31] A. Tharwat, "Independent component analysis: An introduction," *Applied Computing and Informatics*, vol. 17, no. 2, pp. 222–249, Jan. 2020.

[32] L. Turchet, M. Lagrange, C. Rottondi, G. Fazekas, N. Peters, J. Østergaard, F. Font, T. Bäckström, and C. Fischione, "The internet of sounds: Convergent trends, insights, and future directions," *IEEE Internet of Things Journal*, vol. 10, no. 13, pp. 11 264–11 292, 2023.

[33] D. Veitch, S. Babu, and A. Pàsztor, "Robust synchronization of software clocks across the internet," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement*, ser. IMC '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 219–232. [Online]. Available: https://doi.org/10.1145/1028788.1028817

[34] F. Walls and J.-J. Gagnepain, "Environmental sensitivities of quartz oscillators," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, vol. 39, no. 2, pp. 241–249, 1992.