Low-cost synchronization of high-speed audio and video recordings in bio-acoustic experiments

Dennis Laurijssen*, Erik Verreycken*, Inga Geipel‡¶, Walter Daems*, Herbert Peremans†, Jan Steckel*§‖
*CoSys Lab, Faculty of Applied Engineering, University of Antwerp, Belgium †APL, FTEW-ENM department, University of Antwerp, Belgium ‡Smithsonian Tropical Research Institute, Balboa, Ancón, Republic of Panama §Flanders Make Strategic Research Centre, Belgium ¶Wissenschaftkolleg zu Berlin, Berlin, Germany
‖jan.steckel@uantwerpen.be

Summary statement

This work describes a low-cost method to reliably synchronize experimental data streams such as audio, video or other inputs using a 1-bit synchronization signal.

Abstract

In this paper, we present a method for synchronizing high-speed audio and video recordings of bio-acoustic experiments. By embedding a random signal into the recorded video and audio data, robust synchronization of a diverse set of sensor streams can be performed without the need to keep detailed records. The synchronization can be performed using recording devices without dedicated synchronization inputs. We demonstrate the efficacy of the approach in two sets of experiments: behavioral experiments on different species of echolocating bats and the recordings of field crickets. We present the general operating principle of the synchronization method, discuss its synchronization strength and provide insights in how to construct such a device using off-the-shelf components.

I. INTRODUCTION

Data-acquisition in behavioral experiments on a wide range of animal species, often relies on multi-modal (i.e. video and single- or multi-channel audio) sensor information (Valente et al. 2007). Indeed, many behavioral experiments on bats (for example (Geberl et al. 2015, Geipel et al. 2013, Greif et al. 2017, Luo and Moss 2017, Stilz 2017, Ubernickel et al. 2013)), zebra finches (e.g. (Ullrich et al. 2016)) and even insects such as fruit flies (Coen and Murthy 2016) rely on capturing synchronized high-speed video and single- or multi-channel audio data. Having an accurate measure of the relative time-shift between the individual sensor data streams is essential for subsequent analysis of the sensor data. Determining and compensating for the time-shift between multiple sensor data streams is referred to as time synchronization. The time synchronization of both commercial and custom-built multimodal sensor systems often relies on using a so-called trigger event, e.g. a post-trigger which is provided by the experimenter once the animal has performed its task. This finishes the data capture sequence on multiple devices simultaneously providing a synchronization point for all the captures sequences. This is often performed by broadcasting a digital Transistor-Transistor Level (TTL) pulse to the dedicated synchronization inputs of each individual data-acquisition device. We argue that this synchronization approach suffers from two important disadvantages. *First,* this approach depends on the availability of compatible synchronization inputs on the various data-acquisition devices. Furthermore, if the synchronization handling mechanism is not implemented carefully precise synchronization cannot be guaranteed, e.g. whenever part of the synchronization system relies on a software component running on a non real-time operating system. *Secondly,* the downside of the digital synchronization pulse based approach is that the synchronization information is not embedded in the data. By synchronizing either the start or the end of the captured data sequences, the relative time-shift between the individual data streams, e.g. audio and video streams, can be deduced. However, this synchronization information is easily lost in the case of truncation of the data sequences. Furthermore, data-sequences are often recorded in such a way that a portion of the data is recorded before the so-called trigger event, and a portion after the trigger event (pre- and post-trigger data). The information about the type of the captured data sequence needs to be recorded very carefully in meta-data, which increases the risk of data loss or inconsistencies. Again, truncation of the data, i.e. throwing away uninteresting sections of the data sequences, aggravates the risk for inconsistencies.

To overcome these shortcomings of traditional synchronization techniques, we propose a method based on embedding a random 1-bit signal into the data streams directly. This type of signal is exceptionally good for alignment purposes as it exhibits a very narrow autocorrelation function. Embedding this type of synchronization signal into the recorded data sets, solves both issues at once: on the one hand, no specialized synchronization input is needed to store synchronization information and the accuracy and precision of the synchronization do not depend on the manufacturer of the recording equipment. In addition, as the synchronization information is embedded into the data streams directly, the synchronization information can be made very robust to truncation and even re-sampling of the sensor data. It should be noted that our proposed approach
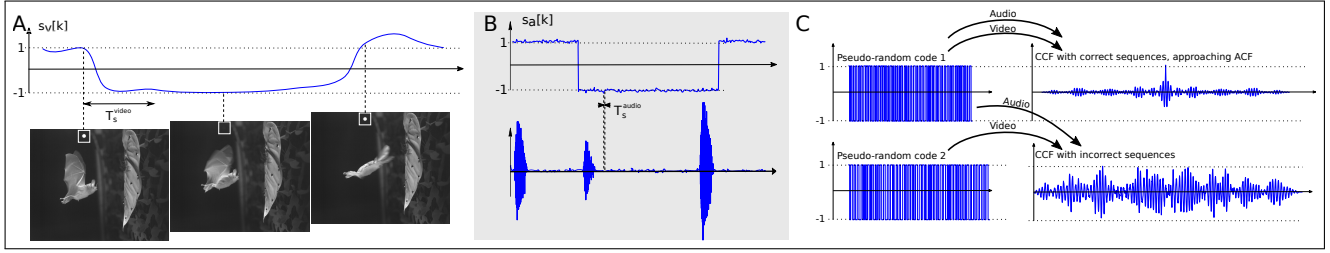
Fig. 1: Panel A illustrates the extraction process of the synchronization signal from a video sequence. The intensity of the region of interest is averaged for each time point and put into a time-vector $s_v[k]$. This signal is then thresholded to obtain the 1-bit synchronization signal. It should be noted that this signal is sampled much slower (between 100 and $5000\,\mathrm{Hz}$) than the audio data (around $500\,\mathrm{kHz}$). Panel B shows the synchronization signal extracted from the audio data in relationship to some bat echolocation calls, sampled at $500\,\mathrm{kHz}$. Panel C, top row, shows the effect of crosscorrelating two sequences that belong together, resulting in a very peaked crosscorrelation function (CCF), approaching the autocorrelation function (ACF) of the pseudo-random sequence. The bottom row illustrates what happens when two pseudo-random sequences are correlated which do not belong together. The crosscorrelation function is smeared out, allowing the detection of wrongly combined measurement pairs.

is similar in concept to the SMPTE timecode system (Poynton 1996) used in synchronization for television audio and video, with as main difference that we propose to embed a random sequence in the data instead of using structured timecode.

II. MATERIALS AND METHODS

A. Topology of the synchronization system

The proposed synchronization system consists of two main parts: a 1-bit signal generator with random periods, and one or more means of transferring the generated random sequences to the sensor modalities that require synchronization. In case of video and multi-channel audio these means are a blinking Light Emitting Diode (LED) on the one hand, and an electrical copy of the 1-bit synchronization signal on the other hand (e.g. a TTL-compatible version with a -1 represented by $0\,\mathrm{V}$, and a +1' represented by $5\,\mathrm{V}$). The blinking LED is recorded using the video camera, and the electrical signal is recorded with the multi-channel microphone recording equipment sacrificing one microphone channel. Indeed, many multi-channel microphone array systems can spare a single channel in return for highly-accurate synchronization with video data.

Discrete signals consisting of 1-bit random sequences have the advantage of exhibiting an autocorrelation function (ACF) that in the limit of infinite duration approaches the (discrete time) *unit impulse* (Oppenheim et al. 1997):

$$\delta[n] = \begin{cases} 0 & \text{if } n \neq 0 \\ 1 & \text{if } n = 0 \end{cases},$$

with n the time lag expressed in number of sample periods. Random sequences, even if only pseudo random, also have very low crosscorrelation function values (CCF) with different instantiations of the same (pseudo-random) process. In addition to being straightforward to generate, this property makes them an excellent choice for the synchronization application proposed in this paper. Indeed, as shown in Fig. 1C (upper panel), very accurate time alignment can be achieved for data sequences that belong together: they exhibit a distinctly peaked

crosscorrelation function. This is further illustrated in Fig. 2A and Fig. 3. On the other hand, audio and video data that were incorrectly assumed to match, can be easily recognized as non corresponding through the lack of a distinct impulse-like crosscorrelation peak, see Fig. 1C (lower panel).

B. Pseudo-random number generators

To implement the proposed synchronization system, a pseudo-random number generator (PRNG) may be used. Often, computing environments offer pseudo-random number generators that generate number sequences based on fixed algorithms or tables. The numbers generated in the sequence exhibit a large entropy, i.e. they appear to be random, though the sequence itself is deterministic (and periodic). However, when implemented properly, the period of these generators will be very large (e.g., the Mersenne Twister random number generator (default in Matlab), exhibits a periodicity of $2^{19937} - 1$ (Matsumoto et al. 1998)). Such a PRNG can be used to generate 1-bit (pseudo-random) sequences. Assuming a 32-bit PRNG, we can derive a 1-bit pseudo random signal by using the 32-bit random values r as values for a timer-scaler to calculate a time period (P_{wait}) lying in between a predefined minimum and maximum period (P_{min} and P_{max}). The synchronization system will wait P_{wait} before toggling the 1-bit output value, thus creating the 1-bit pseudo-random signal. The equation used for calculating the timer wait period is:

$$P_{wait} = P_{min} + \frac{r}{2^{32}}(P_{max} - P_{min})$$

Given this model, the key question is how to pick a good value for P_{min} and P_{max}. The basis for a good selection is summarized in Fig. 4. The conclusions that can be drawn from the figure, lead to the following selection procedure:

1) Select the minimal period P_{min} that the output of the pseudo-random generator remains in the same state
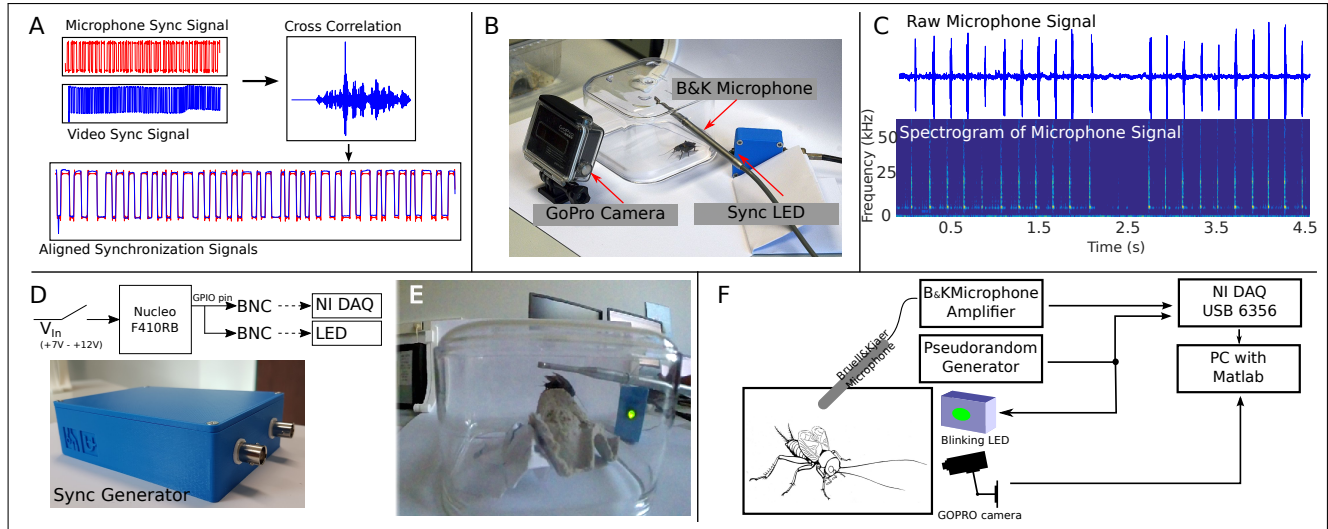
Fig. 2: Overview of the experiment in the lab environment for recording high-speed video and audio from a common field cricket (*Gryllus campestris*). Panel A shows the extracted synchronization signals from the microphone (red) and video (blue) data streams. The cross correlation between the two sequences exhibits a single narrow peak, which can be used to calculate the time shift between the two signals. Using the calculated time shift between the two signals can be used to align the synchronization signals (bottom of panel B). Panel B shows the experimental setup for the high-speed recordings of the field cricket: a GoPro camera running at 240 Hz, a Brüel & Kjær microphone and the synchronization LED (blue box). Panel C shows the recorded microphone signal and spectrogram. Panel D shows the synchronization generator built around a Nucleo F410RB microcontroller. Panel E shows a single frame of the recorded video. Panel F shows the overview of the hardware measurement setup: the B&K microphone is connected via an amplifier to a National Instruments data-acquisition (DAQ) device. The pseudo-random generator is connected to the blinking LED and to the DAQ-device. The data is transferred to a PC running Matlab for further analysis.
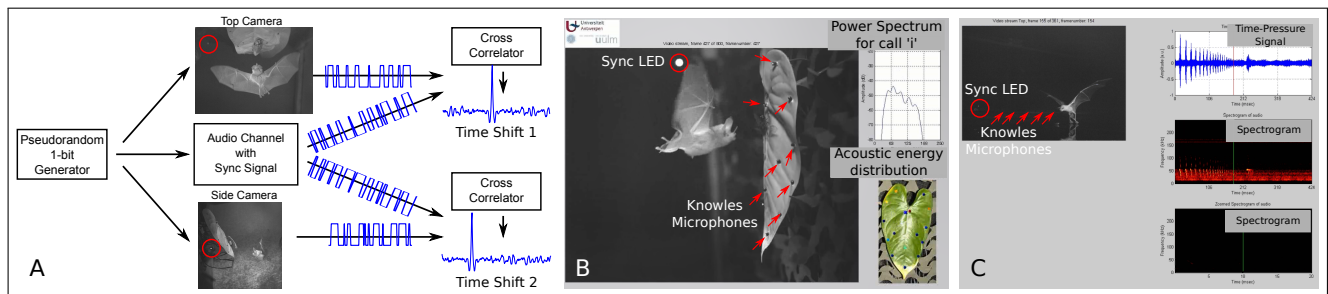

Fig. 3: Panel A shows the overall topology of the synchronization system. A 1-bit pseudo-random generator is coupled either optically through a LED into a video camera or electrically to a multi-channel microphone recording system. Crosscorrelating the synchronization signals recorded by the two sensor modalities, allows calculating the exact time-shift required for aligning the two sensor data sequences. Note also that by placing a separate LED in the field of view of each camera, the synchronization of multiple camera recordings can be easily accommodated. Panel B shows an experimental setup where a *Micronycteris microtis* is capturing a dragonfly on a leaf. Sixteen Knowles FG-type microphones are mounted in the leaf surface (indicated by red arrows), which allows the calculation of the distribution of the acoustic energy on the leaf surface. Video is recorded using a high-speed (500 Hz) infrared camera, and the infrared LED used for synchronization is indicated by a red circle. Panel C shows the experimental setup used to document a *Noctilio leporinus* capturing a fish from a body of water. Again, Knowles FG-type microphones are mounted near the edge of the water surface (indicated by red arrows), and the IR-LED is indicated with a red circle. Furthermore, the panel shows the time-pressure signal of one of the microphone channels, and the corresponding spectrogram.
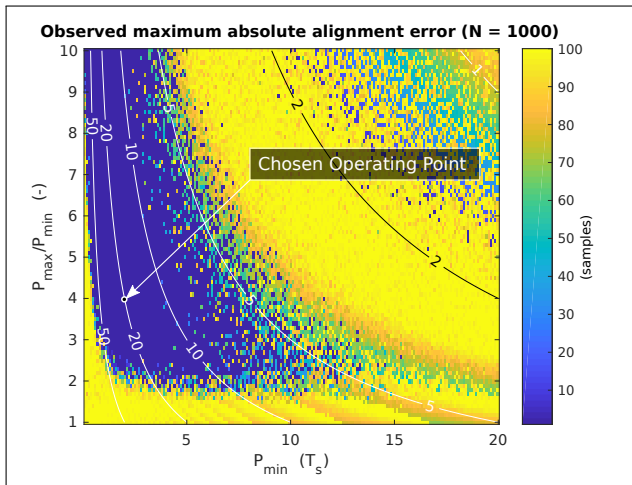
Fig. 4: Maximum alignment error (in number of sample periods) of a 100-sample signal fragment observed over $N = 1000$ simulations (assuming uniform sample desynchronization in $[-T_s/2, T_s/2]$, with T_s the sampling period) as a function of P_{max} and P_{max}/P_{min}. Superimposed on the graph, contours have been drawn indicating the expected number of transitions of the synchronization signal on the total fragment length of 100. As can be clearly seen, choosing $P_{min} \leq 2T_s$ is disadvantageous for it causes transitions to be skipped in the recorded signal fragment. This corresponds to the requirement of obeying Shannon's sampling theorem. In addition, taking P_{max}/P_{min} too low (close to 1) increases the risk of yielding periodic synchronization signals for smaller fragments. Likewise, the combination of P_{min} and P_{max} should be such that on average at least 10 transitions remain in the smallest fragment one wants to consider. The operating point chosen for the example in the text has been indicated on the graph.

(either 1 or 0) to be $P_{min} = 2T_s$ with T_s the largest sampling period (corresponding to the slowest sampling rate) in the entire recording system. This obeys Shannon's sampling requirement to avoid missing transitions in the synchronization signal.

2) Determine the smallest fragment length L (in number of samples) one ever wants select from a signal while still being able to synchronize it with other related signals.

3) Given L, choose P_{max}/P_{min} as large as possible, without reducing the estimated number of transitions below 10. The number of transitions K can be estimated using:

$$K = \frac{2 \cdot L \cdot T_s}{P_{min} \left(\frac{P_{max}}{P_{min}} + 1 \right)}$$

The lower limit of 10 transitions can be considered to be constant with respect to the fragment length. Fig. 5 illustrates the relationship between the number of transitions and the strength of the autocorrelation peak.

Note that in Fig. 4, the total fragment length has been chosen rather short (only 100 samples), leading to a very limited range of proper P_{min} and P_{max} values, and a relative high risk of meeting an odd failure of the synchronization principle. The
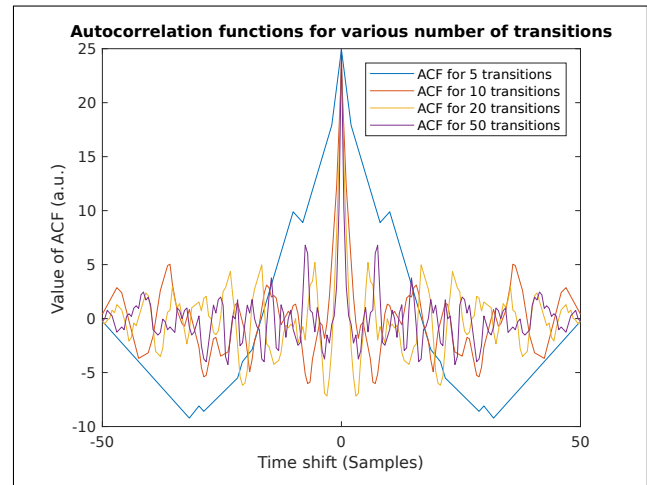

Fig. 5: Illustration of the relationship between the number of transitions in the synchronization signal and the strength of the autocorrelation peak.

length has been specifically chosen short to illustrate these effects. In practical cases, the length of the fragment will be significantly bigger, resulting in a larger valid range and a vanishing chance of a failing synchronization. In practice, we commonly use fragment lengths above 500 samples, easily accommodating for 50 transitions and more.

Let's illustrate the procedure outlined above with a practical example. Most often, the slowest recording modality is a video camera, e.g. even high-speed video cameras typically have measurement rates between only $100\,\mathrm{Hz}$ and $5000\,\mathrm{Hz}$. We choose the minimal period P_{min} to be at least twice the sampling period of the slowest device. For example, for a camera with a sampling rate of $100\,\mathrm{Hz}$, we set the minimal period $P_{min} = 20\,\mathrm{ms}$. The maximal period is chosen to be four times the minimal period: $P_{max} = 80\,\mathrm{ms}$. This allows for a good synchronization of fragments as small as $500\,\mathrm{ms}$.

As hardware for an implementation, an ST-Microelectronics Nucleo F410RB development board was chosen. This development board features an ARM Cortex M4 micro-controller together with a universal serial bus (USB) programmer and debugger. The board can be powered using the USB connection or using an external direct current (DC) voltage (between $7\,\mathrm{V}$ and $12\,\mathrm{V}$). The STM32F410 micro-controller includes a great number of peripherals on chip. The most interesting peripheral for this application is the integrated hardware true-random number generator (TRNG), that generates random 32-bit values using analog noise. This is an improvement over the pseudo-random number generator that has been assumed before, though it is in no way required for a well functioning system. A pseudo-code description of the software running on the micro-controller is presented in Listing 1. For easy replication of the proposed synchronization system, we also provide Arduino-compatible code (Listing 2) which can be easily implemented to build synchronization devices. In order to feed the 1-bit random signal to multiple external measurement systems, the micro controller output pin is connected

to multiple Bayonet Neill-Concelman (BNC) connectors. In the experimental setup shown in Fig. 2, one synchronization output is connected to a National Instruments DAQ-device, acquiring both the microphone signal and the synchronization signal, while the other is connected to a LED driver placed in the field of view of a GoPro camera.

Listing 1: Pseudo-code of an embedded 1-bit pseudo-random generator

```
initialize peripherals;

set output pin state;
generate random wait period;
start timer;

while(1){
  if (timer period elapsed){
    stop timer;
    toggle output pin state;
    generate random wait period;
    start timer;
  }
}
```

Listing 2: Arduino implementation of a 1-bit pseudo-random generator on output pin 13

```
void setup()
{
  pinMode(13, OUTPUT);
}

void loop()
{
  digitalWrite(13, HIGH);
  // Wait for a random period between 20ms and 80ms
  delay(random(20,80));
  digitalWrite(13, LOW);
  // Wait for a random period between 20ms and 80ms
  delay(random(20,80));
}
```

Note that in many cases DAQ devices come with high-speed DA channels or digital output channels that may be used to generate the pseudo-random synchronization signal. In addition, most studio audio devices provide a large number of DA channels that can be used, driving other audio device inputs and even directly driving a LED for video synchronization (in case the output is DC coupled and is able to source a sufficient amount of current). As many such devices support DA conversion rates up to $192\,\mathrm{kHz}$ or higher, the proposed method can be implemented yielding accurate synchronization, without the need to build separate hardware. One only needs to generate an audio sequence according to the principles outlined above.

C. Aligning multiple data sequences

To synchronize the separate data sequences, the extraction of the synchronization signals $s_a[k]$ from the audio signal, and $s_v[k]$ from the video signal is required. Note that usually different sensor modalities will have different sample rates: high-speed video in the order of $100\,\mathrm{Hz}\text{-}5000\,\mathrm{Hz}$, and audio in the order of $8\,\mathrm{kHz}\text{-}500\,\mathrm{kHz}$, depending on the frequency range of interest. Therefore, the extracted synchronization

signals need to be resampled to arrive at a common sampling frequency. For practical reasons, i.e. video data cannot be easily upsampled to the audio sampling frequency for memory efficiency reasons, we chose the lowest sampling frequency as the reference rate in our implementation. The resampling step results in a subsampled audio synchronization signal $s_a^r[k]$. As P_{min} was chosen to be at least three times the sampling period of the slowest device this subsampling will not give rise to aliasing. Next, we calculate the CCF between the two synchronization signals

$$s_c[n] = \sum_k s_a^r[k] \cdot s_v[k+n]$$

for all possible values of the time shift n. Next, we determine the maximum of the cross correlation function $s_c[n]$ and the corresponding time-shift n^* between the audio and the video synchronization signals:

$$n^* = \arg\max_n s_c[n]$$

This time-shift n^* can then be used to align the audio and video sequences.

Extracting the synchronization signal $s_a[k]$ from the audio data is trivial, as it is directly captured by one of the channels of the audio recording equipment. Extracting the video synchronization signal $s_v[k]$ requires a bit of extra image processing. In our implementation, given the video sequence $f_{x,y}[k]$, we select a square region of interest $R_{x,y}$ defined around the position of the LED in the image. Next, we calculate for every frame at timestep k the average intensity in the region $R_{x,y}$:

$$s_v^r[k] = \frac{\sum_{(x,y)\in R_{x,y}} f_{x,y}[k]}{N}$$

with N the number of pixels contained in the region $R_{x,y}$. Finally, we subtract the mean from the raw video signal $s_v^r[k]$ to obtain the video synchronization signal $s_v[k]$:

$$s_v[k] = s_v^r[k] - \frac{\sum_k s_v^r[k]}{M}$$

with M the length of the raw video synchronization signal $s_v^r[k]$. The extraction of the synchronization signal is illustrated in Fig. 1A and 1B.

III. RESULTS AND DISCUSSION

A. Illustration: cricket songs

To illustrate the efficacy of the proposed approach we have performed two sets of experiments. The first experiment consists of recording a calling field cricket (*Gryllus campestris*) using a GoPro Hero 3+ camera capturing video at 240 frames per second. The audio was recorded using a Brüel & Kjær 1/8th inch microphone, and the signals were recorded using a National Instruments USB 6356 data-acquisition (DAQ) device. Fig. 2 shows the setup in more detail. We performed audio and video recordings of 5 seconds, and extracted the synchronization signals from audio and video separately. Using custom-made Matlab code we performed the alignment of the audio and

video data. Separate wave (audio) and MPEG-4 (video) files were written to the harddisk using a 10x lower sampling rate. Using a video-editing tool (Wondershare Filmora) both sequences were combined, and the resulting video is shown in video S1. When observing the video, the motion of the cricket's wings during stridulation is synchronized with the recorded sound.

B. Illustration: bat behavioral experiments

A second illustration is a series of bat behavioral experiments performed during the EU-FP7 project 'Chiroping', on Barro Colorado Island, Panama. We performed experiments on *Micronycteris microtis*, *Macrophyllum macrophyllum* and *Noctilio leporinus*, a gleaning and two trawling bats respectively. All required research permissions were obtained from the Panamanian Environmental Agency (ANAM) and the Smithsonian Institutional Animal Care and Use Committee (IACUC; 2008-11-06-24-08). The acoustic data was recorded with a custom-made 16-channel microphone array based around Knowles FG-23329-p07 condenser microphones. The video data was recorded with a high-speed camera (Cam-Record CR600x2, Optronis GmbH, Germany) at 500 frames per second (*M. microtis*) and 250 frames per second (*N. leporinus*) respectively. Synchronization was performed using our proposed synchronization mechanism, and audio and video data were combined in single video files. The results of these measurements are shown in video S1 and annotated screenshots of the *M. microtis* and *N. leporinus* recordings can be seen in Fig. 3.

C. Discussion

In this paper, we have described and demonstrated a flexible and low-cost synchronization method for multi-modal bio-acoustic experimental data. Our proposed synchronization method relies on embedding synchronization information into the sensor data directly. The main advantages are (1) that no manufacturer provided synchronization method is needed, (2) that synchronization information is not easily lost, i.e. no manual recording of meta-data is required, and (3) different sensor data streams can be easily checked for correspondence. As the method does not rely upon manufacturer standardized synchronization mechanisms, it can be easily extended to other sensing modalities using vibration sensors, force sensors, etc., by electrically coupling them into a sacrificial channel of a multi-channel recording device. The synchronization method can even be extended to synchronize with optical 3D tracking equipment, e.g. the Qualisys Miqus cameras, by using an 830 nm infrared LED. We have provided an example of how to construct such a synchronization device using off-the-shelf hardware components, and provided a pseudo-code implementation for the pseudo-random generator.

Currently we have only demonstrated synchronization with multi-channel recording systems through a sacrificial data-channel. In the case were multi-channel recording equipment

is not available, other means of inserting the synchronization data can be devised. For example, the synchronization information could be embedded in the least-significant bit (LSB) of the recorded data, which is often occupied by noise in real-world recordings. This, however, would require alteration to the recording hardware, making this approach less straightforward. The synchronization information might also be inserted acoustically by using Amplitude-Shift Keying (ASK) modulation (or more advanced modulation schemes) in a section of the acoustic spectrum which is not relevant to the biological experiment, resembling the approach used in the cameras using a blinking LED. This ASK-modulated signal can also be inserted electrically into the Analog-to-Digital Converter (ADC) of the recording device, requiring a small electronic circuit to sum the microphone signal with the ASK-modulated signal.

The proposed approach opens the opportunity to synchronize an arbitrary number of data-acquisition systems and sensor streams, as no intrinsic limitation is present in the proposed architecture. During our bat behavioral experiments, we routinely synchronized up to four high-speed cameras with two 16-channel microphone arrays. We argue that the flexibility and robustness of our proposed approach, in combination with the fact that it can be constructed using off-the-shelf components, makes it a useful tool that can be applied in a broad range of biological behavioral experiments in which the combined recording of multi-sensor data-streams is required.

COMPETING INTERESTS

The authors declare no competing interest. The funder had no role in study design.

AUTHOR CONTRIBUTIONS

Development of synchronization hardware: DL, EV, JS, HP, WD. Performing bio-acoustic experiments: DL, IG, JS, HP. Writing the manuscript: EV, JS, DL, IG, HP, WD.

LIST OF ABBREVIATIONS

Abbreviation	Description
ACF	autocorrelation function
ANAM	Panamanian environmental agency
ASK	amplitude shift keying
BNC	bayonet Neill-Concelman
CCF	crosscorelation function
DAQ	data acquisition
DC	direct current
EU-FP7	seventh framework programme of the European Union
IACUC	institutional animal care and use committee
LED	light-emitting diode
LSB	least-significant bit
MPEG	motion pictures expert group
PRNG	pseudo-random number generator
SMPTE	society of motion picture and television engineers
TRNG	true-random number generator
TTL	transistor-transistor level
USB	universal serial bus

REFERENCES

Coen, P. and Murthy, M. (2016), 'Singing on the fly: sensorimotor integration and acoustic communication in Drosophila', *Current Opinion in Neurobiology* **38**, 38–45.
   URL: *http://www.sciencedirect.com/science/article/pii/S0959438816000258*

Geberl, C., Brinkløv, S., Wiegrebe, L. and Surlykke, A. (2015), 'Fast sensory-motor reactions in echolocating bats to sudden changes during the final buzz and prey intercept.', *Proceedings of the National Academy of Sciences of the United States of America* **112**(13), 4122–7.
   URL: *http://www.ncbi.nlm.nih.gov/pubmed/25775538 http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC4386384*

Geipel, I., Jung, K. and Kalko, E. K. V. (2013), 'Perception of silent and motionless prey on vegetation by echolocation in the gleaning bat Micronycteris microtis.', *Proceedings. Biological sciences* **280**(1754), 20122830.
   URL: *http://www.ncbi.nlm.nih.gov/pubmed/23325775 http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3574334*

Greif, S., Zsebk, S., Schmieder, D. and Siemens, B. M. (2017), 'Acoustic mirrors as sensory traps for bats', *Science* **1047**(September), 1045–1047.
   URL: *http://science.sciencemag.org/content/357/6355/1045/tab-pdf*

Luo, J. and Moss, C. F. (2017), 'Echolocating bats rely on audiovocal feedback to adapt sonar signal design', *Proceedings of the National Academy of Sciences* p. 201711892.
   URL: *http://www.pnas.org/content/early/2017/09/20/1711892114.full?casa_token=18YAgSiZfFcAAAAA:LmutIQ9rwY6HEqVS-cGmgp0Cypp-8DFuVe24qvcGL9M8ZRFInt6OoLkPzXm4lp5-7B6BDCmnbGT2kg*

Oppenheim, A. V., Willsky, A. S. and Nawab, S. H. (1997), *Signals & systems*, Prentice Hall.

Poynton, C. A. (1996), *A technical introduction to digital video*, J. Wiley.

Matsumoto, M., and T. Nishimura (2007), 'Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudorandom Number Generator.', *ACM Transactions on Modeling and Computer Simulation*, 8(1):3-30.

Stilz, P. (2017), 'How glass fronts deceive bats', *Science* **357**(6355).
   URL: *http://science.sciencemag.org/content/357/6355/977.full*

Ubernickel, K., Tschapka, M. and Kalko, E. K. V. (2013), 'Flexible echolocation behavior of trawling bats during approach of continuous or transient prey cues.', *Frontiers in physiology* **4**, 96.
   URL: *http://www.ncbi.nlm.nih.gov/pubmed/23675352 http://www.pubmedcentral.nih.gov/articlerender.fcgi?artid=PMC3650317*

Ullrich, R., Norton, P. and Scharff, C. (2016), 'Waltzing Taeniopygia : integration of courtship song and dance in the domesticated Australian zebra finch', *Animal Behaviour* **112**, 285–300.
   URL: *http://linkinghub.elsevier.com/retrieve/pii/S000334721500425X*

Valente, D., Wang, H., Andrews, P., Mitra, P. P., Saar, S., Tchernichovski, O., Golani, I. and Benjamini, Y. (2007), 'Characterizing Animal Behavior through Audio and Video Signal Processing', *IEEE Multimedia* **14**(4), 32–41.
   URL: *http://ieeexplore.ieee.org/document/4354155/*