Audio Engineering Society

Convention e-Brief 555

Presented at the 147th Convention
2019 October 16 – 19, New York

This Engineering Brief was selected on the basis of a submitted synopsis. The author is solely responsible for its presentation, and the AES takes no responsibility for the contents. All rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Audio Engineering Society.

Realistic procedural sound synthesis of bird song using particle swarm optimisation.

Jorge Zúñiga[1] and Joshua D. Reiss[1]

[1]*Queen Mary University of London*

Correspondence should be addressed to Joshua D. Reiss (`joshua.reiss@qmul.ac.uk`)

ABSTRACT

We present a synthesis algorithm for approximating bird song using particle swarm optimization to match real bird recordings. Frequency and amplitude envelope curves are first extracted from a bird recording. Further analysis identifies the presence of even and odd harmonics. A particle swarm algorithm is then used to find cubic Bezier curves which emulate the envelopes. These curves are applied to modulate a sine oscillator and its harmonics. The synthesised syllable can then be repeated to generate the sound. 36 bird sounds have been emulated this way, and a real-time web-based demonstrator is available, with user control of all parameters. Objective evaluation showed that the synthesised bird sounds captured most audio features of the recordings.

1 Introduction

Most recent academic research on bird sounds deals with the problem of identifying bird species based on a sound recording, e.g. [1]. Species identification through audio has many practical uses including supporting the census of population sizes, improving taxonomy and even preventing bird collisions with airplanes. The main problems are dealing with environment noise and choice of audio features. There are also different methods for modelling and segmenting the sound. The most widely used consists of segmenting and isolating each syllable of a bird's call e.g. [2].

Bird song can be very complex and vary widely. Birds can have several versions of their song, known as song type; from the low phrase vocabulary, noisy and frequency dense caws of a crow to the high phrase vocabulary flute-like tones of a blackbird. High vocabulary makes the problem of identifying species more difficult [3]. They convey complex behaviours in the way they are influenced and influence others with calls and exhibit consistent temporal interaction patterns [4].

A good bird song synthesis model should be able to reproduce a wide variety of high quality vocalizations for a particular and across species.

Catchpole and Slater [5] define syllables as a series of units which occur together in a particular pattern. Each syllable comes from a species dependent vocabulary, with different degrees of similarity depending on the species. These syllables are grouped in phrases which are the sections in the song or call. Complex syllables can also be segmented into elements or notes. An element is defined as a continuous line on a spectrogram.

An important set of features necessary when studying bird song is related to frequency modulation. Stowell and Plumbley [6] find that simple frame-to-frame peak energy bins works well as features for large scale bird classification tasks.

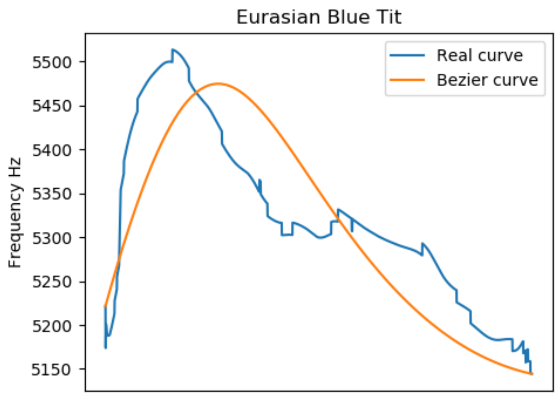

Fig. 1: Comparison of the Bezier curve approximation and the fundamental frequency of a Blue Tit.

Despite the interest in bird song within the wider field of Computational Bioacoustic Scene Analysis [7], there has been little work on synthesising such sounds. Human speech synthesis has served as an interesting starting point for other kinds of animal sounds. Anikin [8] and [9] provided systems for synthesizing non-verbal vocalizations and both were easily be adapted to produce nonhuman animal sounds such as cats and wolves.

But birds have a very different vocal system, and hence a different approach was taken here. The synthesis model presented in this paper imitates sounds consisting of single, repeated syllables. The model is most suited to simpler bird vocalizations where the syllable is also a note and is repeated.

We introduce a method for learning the modulator's shape required to synthesize the syllable using particle swarm optimization. The rest of the parameters required for synthesis are approximated by different methods, namely the length and rate of the syllables and two parameters describing the quality of the harmonics, whether they are odd or odd and even, and the amplitude ratio between the first two harmonics.

A javascript interactive version of the model with presets for 36 birds is available as a JSAP plug-in [10] on FXive. FXive [11] is a real-time sound effect synthesis browser framework at `https://fxive.com/app/main-panel/real-birds.html`. It supports a large number of procedural audio models, such as impact sounds [12], environmental sounds [13] and acoustic phenomena [14] .

2 Dataset

Bird sounds used for testing and learning came from an open database of bird calls and song, `https://www.xeno-canto.org/` [15]. It contains bird recordings from around the world of over 10000 species. Recordings of various birds were gathered from `https://www.kaggle.com/rtatman/british-birdsong-dataset/version/2`, a small dataset of 88 species from the UK. This dataset was chosen because it contains manually curated good quality recordings which are sometimes hard to come by on a public database such as xeno-canto. Species from the dataset were selected on the criteria of it being a repeating syllable sound composed of harmonics of short bandwidth.

Two files were manually created from each recording; a file of the single syllable cut perfectly on both ends and a file containing three syllables. The three syllable file was used to compute the syllable rate. The single syllable file was used to produce both fundamental frequency curve and amplitude envelope curves as well as the other parameters.

3 Model

Bezier curves are parametric curves described by points. They were originally used to design the bodywork of cars but are now used extensively in computer graphics and animation [16]. The curves can have any number of control points. Here, we use cubic Beziers, which consist of four control points, to describe the frequency curve and the amplitude envelope of a bird syllable. They are computed from the control points and then used as modulators on a sine oscillator. An example of approximating a fundamental frequency curve is shown in Figure 1.

Additional oscillators are added on top of the fundamental to implement overtones. The amplitude curve is applied to each one with a successively lower amplitude described by an attenuation parameter ranging from 0 to 1, and the frequency curve is multiplied by the harmonic number. All computed harmonics that exist below the Nyquist frequency are added to produce the final sound. The model can synthesize odd, or odd and even harmonics selected by a Boolean parameter. The last two parameters are rate and length. Rate is in Hertz and describes how fast the syllable repeats itself. Length is in seconds and describes the syllable duration. Table 1 shows the list of parameters.

Table 1: Parameter list.

Learnt parameters	Computed Parameters
Frequency control 2 X	Start frequency
Frequency control 2 Y	End frequency
Frequency control 3 X	Rate
Frequency control 3 Y	Length
Amplitude control 2 X	Overtone attenuation
Amplitude control 2 Y	Harmonics are odd
Amplitude control 3 X	
Amplitude control 3 Y	

4 Learning

4.1 Extracting parameters

The first and last control points of a Bezier lie on the curve. So they describe the start and end frequencies for the frequency curve, and are set to 0 for the amplitude curve. The parameters which are learned are the *x* and *y* values of the two middle control points of the amplitude and frequency Beziers.

Pitch extraction works by performing an FFT and assuming the highest amplitude bin to be the fundamental frequency. The precise frequency value is then calculated by comparing the expected and measured phase shift between adjacent frames of the chosen bin, as described by Brown and Puckette [17]. This calculated error enables us to find the true frequency value. The window length used was 128 samples and to optimize accuracy we used a hop size of 1 sample. Our implementation uses the aubio library in python [18] with the fcomb preset.

The amplitude envelope curve is found by applying a moving average filter to the absolute value of the signal. The filter uses a 7 ms window. After filtering the fully rectified signal we normalize the result from 0 to 1 to get the resulting curve.

Results from pitch and envelope extraction methods are illustrated in Figures 3 and 4, where we show the spectrogram and waveform of a syllable synthesised with the raw extracted signals and compare it with the Bezier approximation and original recording.

Rate is found by finding the first peak in the self correlation of the three syllable file's amplitude envelope. After self correlating, the middle peak defines the location where the signal was perfectly on top of itself. The distance from the middle peak to the next peak tells us the number of samples a syllable period takes. The

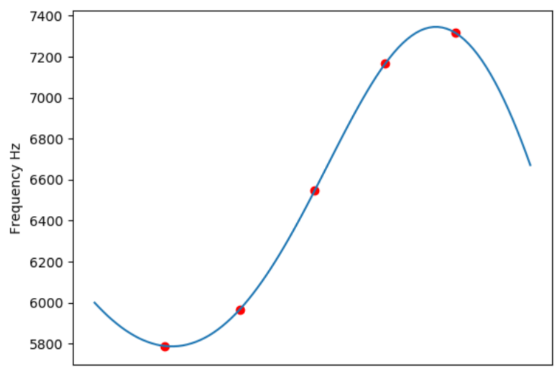

Fig. 2: Points used for learning on a frequency curve of a Worm Eating Warbler approximated by Bezier curves.

syllable length is found by finding the length of a burst of energy from the amplitude envelope.

The overtone attenuation parameter is found by first taking the RMS of the signal. At the point where the RMS is highest, an FFT is taken from a small window on that point. The highest peak in the FFT is assumed to be the fundamental frequency. A peak after that would be the first overtone. The overtone attenuation would be the ratio of the value of the second harmonic and the value of the first harmonic. This tells us how much to attenuate at each overtone. If only one peak is found, the overtone attenuation is set to 0 so there is only one frequency in the synthesised sound.

After taking the FFT of the point with the highest RMS, we determine whether harmonics are odd or both odd and even by finding the distance between the highest peak and the peak after that in the frequency representation of that bin. If the distance is close to the frequency value of the highest peak, assumed to be the fundamental frequency, we know harmonics may be odd and even. Otherwise, harmonics are assumed to be odd.

4.2 Learning parameters

The proposed method of finding the frequency and amplitude control points uses particle swarm optimization. Only the location of the two middle control points for each Bezier needs to be found so the search space is four dimensions, two for the *x* values and two for the *y* values. The *y* values are limited from zero to the nyquist frequency and the x values are from zero to one. The position of a particle describes the shape of the Bezier. The cost on each iteration is calculated by

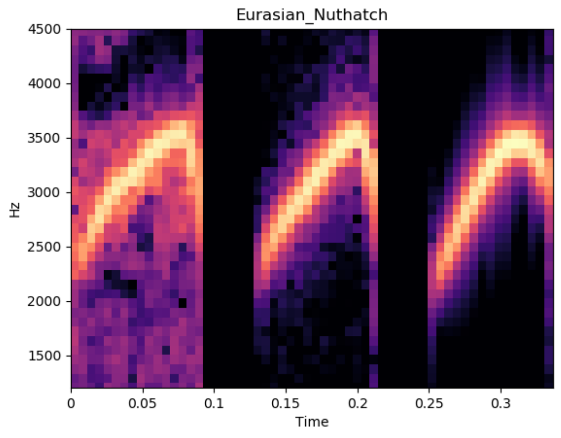

Fig. 3: Comparison spectrogram of a syllable at different stages. Real recording (left), synthesised with extracted curves (center), and synthesised with learnt Bezier approximation curves (right) of an Eurasian Nuthatch.

comparing the curves extracted from the recording to the generated Beziers. The comparison is done by taking five equally spaced values along the curves (shown in Figure 2), excluding the ends, and taking the euclidean distance of both vectors. The swarm algorithm attempts to minimize the cost by iteratively running the objective function on a set of particles.

5 Evaluation

Various audio features were extracted from 36 single syllable recording of different species. The features were also extracted from a syllable synthesised with the learnt parameters and from noise for use as a control variable. For each species, the absolute difference was taken of the real feature and the synthesised feature. The absolute difference was also taken from the real feature and the noise feature. Table 2 shows that the difference in feature values is significantly lower for the synthesised vs real than the real vs noise. A comparison was also done of the resulting cost of the amplitude learning versus the frequency learning.

6 Discussion

Frequency curves (μ 0.122151) were on average more accurately represented than amplitude curves (μ 0.207206) with a similar standard deviation (σ ~0.169). A possible reason for this is that frequency Bezier end points were matched to the frequency curve end points, whereas the amplitude bezier end points

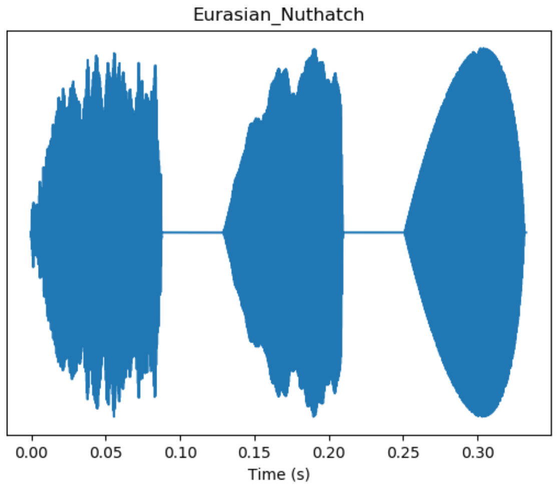

Fig. 4: Comparison waveform of a syllable at different stages. Real recording (left), synthesised with extracted curves (center), and synthesised with learnt Bezier approximation curves (right) of an Eurasian Nuthatch.

were both matched to zero. In some sounds whose syllables were close to each other the amplitude envelope endpoints were higher than zero so when learning the Bezier had a disadvantage having endpoints that were different.

Learning produces varying results. Species with syllables that consist of discrete and powerful bursts produce the best results. Specifically where a syllable consists of a single note. This way, when the frequency curve is initially computed it will be accurate enough so the learning has a stable basis to work with.

Songs that produce bad results are ones that consist of multiple notes. a single Bezier can not approximate a quick jump to another frequency mid syllable. Also, these kinds of sounds generally have a low syllable rate so the differences in frequency of the real and synthesised sound are easier to listen to.

Best results of amplitude curve cost were sounds that had separated syllables in a song. This helped learn a more accurate amplitude envelope.

The synthesis model is limited to bird sounds that consist of repeating syllables. It also can not replicate high bandwidth noise-like sounds, syllables with multiple notes, and long and complex notes. This model, however, produces accurate results for the kinds of sounds it aims to reproduce.

Table 2: Mean and standard deviation of real versus synthesized and real versus noise comparisons on different features.

Feature		Synthesized	Noise
Spectral Centroid	μ	897.073	3452.44
	σ	474.101	751.174
Spectral Bandwidth	μ	744.659	2259.83
	σ	308.257	302.082
Spectral Flatness	μ	0.374674	0.54145
	σ	0.190711	0.0170464
Spectral Flux	μ	43.2346	1067.15
	σ	39.7388	79.3703
Spectral Contrast	μ	6.00999	8.70942
	σ	3.94991	2.01772
Variance	μ	11.7458	14.761
	σ	13.0024	17.9724

7 Further Work

Different curve fittings, such as splines, could be tried to model the syllable, but it is not clear whether a more complex curve would improve the model. The model should also produce a larger range of bird sounds. An interesting bird sound that is difficult to model is the high bandwidth sound which has a noise-like quality. An example of a bird like this is the crow.

A model that learns a bird's different phrases may produce a more varying and interesting sound to listen to for longer periods. A model of this kind would reproduce a species' probability of singing a specific phrase. It would play different phrases at natural rates and times for that species.

References

[1] Stowell, D. et al., "Automatic acoustic detection of birds through deep learning: the first Bird Audio Detection challenge," *Methods in Ecology and Evolution*, 10(3), pp. 368–380, 2019.

[2] Neal, L. et al., "Time-frequency segmentation of bird song in noisy acoustic environments," in *IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2012–2015, 2011.

[3] Stowell, D. et al., "Automatic acoustic identification of individuals in multiple species: improving identification across recording conditions," *J. Royal Society Interface*, 16(153), 2019.

[4] Stowell, D., Gill, L., and Clayton, D., "Detailed temporal structure of communication networks in groups of songbirds," *Journal of the Royal Society Interface*, 13(119), p. 20160296, 2016.

[5] Catchpole, C. and Slater, P., *Bird song: biological themes and variations*, Cambridge university press, 2003.

[6] Stowell, D. and Plumbley, M., "Large-scale analysis of frequency modulation in birdsong data bases," *Methods in Ecology and Evolution*, 5(9), pp. 901–912, 2014.

[7] Stowell, D., "Computational bioacoustic scene analysis," in *Computational analysis of sound scenes and events*, pp. 303–333, Springer, 2018.

[8] Anikin, A., "Soundgen: An open-source tool for synthesizing nonverbal vocalizations," *Behavior Research Methods*, 51(2), pp. 778–792, 2019, doi:10.3758/s13428-018-1095-7.

[9] Wilkinson, W. and Reiss, J. D., "A Synthesis Model for Mammalian Vocalization Sound Effects," in *61st Audio Engineering Society International Conference: Audio for Games*, 2016.

[10] Jillings, N. et al., "JSAP: A plugin standard for the web audio api with intelligent functionality," in *Audio Engineering Society Conv. 141*, 2016.

[11] Bahadoran, P. et al., "Fxive: A web platform for procedural sound synthesis," in *Audio Engineering Society Conv. 144*, 2018.

[12] Sánchez, P. and Reiss, J., "Real-time synthesis of sound effects caused by the interaction between two solids," in *Audio Engineering Society Conv. 146*, 2019.

[13] Moffat, D. and Reiss, J., "Perceptual Evaluation of Synthesized Sound Effects," *ACM Transactions on Applied Perception*, 15(2), p. 13, 2018.

[14] Selfridge, R. et al., "Creating Real-Time Aeroacoustic Sound Effects Using Physically Informed Models," *Journal of the Audio Engineering Society*, 66(7/8), pp. 594–607, 2018.

[15] Vellinga, W.-P. and Planqué, R., "The Xeno-canto Collection and its Relation to Sound Recognition and Classification." in *CLEF (Working Notes)*, 2015.

[16] Farin, G. et al., *Handbook of computer aided geometric design*, Elsevier, 2002.

[17] Brown, J. and Puckette, M., "A high resolution fundamental frequency determination based on phase changes of the Fourier transform," *J. Acoustical Society of America*, 94(2), pp. 662–67, 1993.

[18] Brossier, P. et al., "aubio/aubio: 0.4.9," 2019, doi:10.5281/zenodo.2578765.