Digitizing Buzzing Signals into A440 Piano Note Sequences and Estimating Forager Traffic Levels from Images in Solar-Powered, Electronic Beehive Monitoring

Vladimir A. Kulyukin, Myles Putnam, and Sai Kiran Reka

Abstract—Four multi-sensor, solar-powered, electronic beehive monitoring devices were assembled and deployed over extended periods at two apiaries in Northern Utah to collect 28 gigabytes of audio, temperature, and image data in different weather conditions. An audio processing algorithm is presented for digitizing bee buzzing signals into A440 piano note sequences. The note range detected by the algorithm on a sample of 3421.52 MB of wav data contained the first four octaves, with the lowest note being A0 and the highest note being F4#. The detected notes exhibited a cyclic pattern during a 24-hour period. A computer vision algorithm is proposed for estimating forager traffic levels from images. On a sample of 378 images from a deployed beehive monitoring device, the algorithm achieved an accuracy of 73 percent.

Index Terms—audio analysis, computer vision; electronic beehive monitoring, sustainable computing

I. INTRODUCTION

Since 2006 honeybees have been disappearing from many amateur and commercial apiaries. This trend has been called the colony collapse disorder (CCD) [1]. The high rates of colony loss threaten to disrupt the world's food supply. A consensus is emerging among researchers and practitioners that electronic beehive monitoring (EBM) can help extract critical information on colony behavior and phenology without invasive beehive inspections [2]. Continuous advances in electronic sensor and solar harvesting technologies make it possible to transform apiaries into ad hoc sensor networks that collect multisensor data to recognize bee behavior patterns.

In this paper, algorithms are presented for digitizing buzzing signals into A440 piano note sequences and for estimating forager traffic levels from images. When viewed as time series, note sequences and forager traffic estimates can be correlated with other timestamped data for pattern recognition. It is probable that other musical instruments can be used for obtaining note sequences so long as their notes have standard frequencies detectable in a numerically stable manner.

The standard modern piano keyboard is called the A440 88-keyboard, because it has eighty-eight keys where the fifth A, called A4, is tuned to a frequency of 440 Hz. The standard list of frequencies for an ideally tuned piano is used for tuning actual instruments. For example, A4#, the 50th key on the 88-key keyboard has a frequency of 466.14 Hz. In this paper, the terms *note* and *key* are used interchangeably.

Buzzing signals and images are captured by a solarpowered, electronic beehive monitoring device (EBMD), called BeePi. BeePi is designed for the Langstroth hive [3] used by many beekeepers worldwide. Four BeePi EBMDs were assembled and deployed at two Northern Utah apiaries to collect 28 gigabytes of audio, temperature, and image data in different weather conditions. Except for drilling narrow holes in inner hive covers for temperature sensor and microphone wires, no structural hive modifications are required for deployment.

The remainder of this paper is organized as follows. In Section II, related work is reviewed. In Section III, the hardware and software details of BeePi are presented and collected data are described. In Section IV, an algorithm is proposed for digitizing buzzing signals into A440 piano note sequences. In Section V, the results are presented of applying the digitization algorithm to 3421.52 MB of collected wav data. In Section VI, a computer vision algorithm is outlined for estimating forager traffic levels from images. The algorithm is evaluated on a sample of 378 images. In Section VII, conclusions are drawn.

II. RELATED WORK

Beehives of all sorts and shapes have been monitored by humans for centuries. Gates collected hourly temperature measurements from a Langstroth beehive in 1914 [4]. In the 1950's, Woods placed a microphone in a beehive [5] and identified a warbling noise in the range from 225 to 285 Hz. Woods subsequently built Apidictor, an audio beehive monitoring tool. Bencsik [6] equipped several hives with accelerometers and observed increasing amplitudes a few days before swarming, with a sharp change at the point of swarming. Evans [7] designed Arnia, a beehive monitoring system that uses weight, temperature, humidity, and sound.

Manuscript received December 5, 2015; revised December 24, 2015.

Vladimir. A. Kulyukin is with the Department of Computer Science of Utah State University, Logan, UT 84322 USA (phone: 434-797-2451; fax: 435-791-3265; e-mail: vladimir.kulyukin@usu.edu).

Myles Putnam is with the Department of Computer Science of Utah State University, Logan, UT USA 84322.

Sai Kiran Reka is with the Department of Computer Science of Utah State University, Logan, UT USA 84322.



Fig 1. BeePi hardware components in a Langstroth super.

The system breaks down hive sounds into flight buzzing, fanning, and ventilating and sends SMS or email alerts to beekeepers.

Several EBM projects have focused on swarm detection. S. Ferrari et al. [8] assembled an ad hoc system for monitoring swarm sounds in beehives. The system consisted of a microphone, a temperature sensor, and a humidity sensor placed in a beehive and connected to a computer in a nearby barn via underground cables. The sounds were recorded at a sample rate of 2 kHz and analyzed with MATLAB and Cool Edit Pro. The researchers monitored three beehives for 270 hours and observed that swarming was indicated by an increase of the buzzing frequency at about 110 Hz with a peak at 300 Hz when the swarm left the hive. Another finding was that a swarming period correlated with a rise in temperature from 33° C to 35° C with a temperature drop to 32° C at the actual time of swarming.

Rangel and Seeley [9] investigated signals of honeybee swarms. Five custom designed observation hives were sealed with glass covers. The captured video and audio data were monitored daily by human observers. The researchers found that approximately one hour before swarm exodus, the production of piping signals gradually increased and ultimately peaked at the start of the swarm departure.

Meikle and Holst [10] placed four beehives on precision electronic scales linked to data loggers to record weight for over sixteen months. The researchers investigated the effect of swarming on daily data and reported that empty beehives had detectable daily weight changes due to moisture level changes in the wood.

III. SOLAR-POWERED ELECTRONIC BEEHIVE MONITORING

A. Hardware

A fundamental objective of the BeePi design is reproducibility: other researchers and practitioners should be able to replicate our results at minimum cost and time commitments. Each BeePi consists of a raspberry pi computer, a miniature camera, a solar panel, a temperature sensor, a battery, a hardware clock, and a solar charge controller.

The exact BeePi hardware components are shown in Fig. 1. We used the Pi Model B+ 512MB RAM models, Pi T-Cobblers, half-size breadboards, waterproof DS18B20 digital temperature sensors, and Pi cameras. For solar harvesting, we used the Renogy 50 watts 12 Volts

ISBN: 978-988-19253-8-1 ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online) monocrystalline solar panels, Renogy 10 Amp PWM solar charge controllers, Renogy 10ft 10AWG solar adaptor kits, and the UPG 12V 12Ah F2 sealed lead acid AGM deepcycle rechargeable batteries. All hardware fits in a shallow super, except for the solar panel that is placed on top of a hive (see Fig. 3) or next to it (see Fig. 4).



Fig. 2. Covered BeePi camera.



Fig. 3. Solar panels on beehives.

Two holes were drilled in the inner hive cover under a super with the BeePi hardware for a temperature sensor and a microphone. The temperature sensor chord was lowered into the second deep super (the first deep super is the lowest one) with nine frames of live bees to the left of frame 1. The microphone chord was lowered into the second deep super to the right of frame 9. More holes can be drilled if the placements of the microphone and temperature sensors should be changed.

The camera is placed outside to take static snapshots of the beehive's entrance, as shown in Fig. 2. A small piece of hard plastic was placed above the camera to protect it from the elements. Fig. 3 displays the solar panels on top of two hives equipped with BeePi devices at the Utah State University Organic Farm. The solar panels were tied to the hive supers with bungee cords.

B. Software

In each BeePi, all data collection is done on the raspberry pi computer. The collected data is saved on a 25G sdcard inserted into the pi. Data collection software is written in Python 2.7. When the system starts, three data collection threads are spawned. The first thread collects temperature readings every 10 minutes and saves them into a text file.

The second thread collects 30-second wav recordings every 15 minutes. The third thread saves PNG pictures of the beehive's landing pad every 15 minutes.

A cronjob monitors the threads and restarts them after hardware failures. For example, during a field deployment the camera of one of the EBMDs stopped functioning due to excessive heat. The cronjob would periodically restart the PNG thread until the temperature went down and the camera started functioning properly again.

C. Field Deployment

Three field deployments of BeePi devices have been executed so far. The first deployment was on private property in Logan, UT in early fall 2014. A BeePi was placed into an empty hive and ran exclusively on solar power for two weeks.



Fig. 4. BeePi in an overwintering beehive.

The second deployment was in Garland, UT in December 2014 – January 2015 in subzero temperatures. A BeePi in a hive with overwintering bees is shown in Fig. 4. Due to strong winter winds typical for Northern Utah, the solar panel was placed next to the hive on an empty super and tied down to a hive stand with bungee cords to ensure its safety. The BeePi successfully operated for nine out of the fourteen days of deployment exclusively on solar power. Over 3 gigabytes of pictures, wav files, and temperature readings were obtained during the nine operational days. These field deployments indicate that electronic beehive monitoring may be sustained by solar power.

IV. DIGITIZING BUZZING SIGNALS

Digitizing buzzing signals into A440 piano note sequences is a method of obtaining a symbolic representation of signals grounded in time. When viewed as a time series, such sequences can be correlated with other timestamped data such as estimates of forager bee traffic levels or temperatures.

Bee buzzing audio signals were first processed with the Fast Fourier Transform (FFT), as implemented in Octave [11], to compute the FFT frequency spectrograms to identify the A440 piano key notes. The quality of the computed spectrograms was inadequate for reliable note identification due to low volumes.

The second attempt, which also relied on Fourier analysis, did not use the FFT. A more direct, although less

ISBN: 978-988-19253-8-1 ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online) efficient, method outlined in Tolstov's monograph [12] was used. Tolstov's method relies on periodic functions f(x) with a period of 2π that have expansions given in (1). Multiplying both sides of (1), separately, by cos(nx) and by sin(nx), integrating the products from $-\pi$ to π , and regrouping, the equations in (2) are obtained for the Fourier coefficients in the Fourier series of f(x). The interval of integration can be defined not only on $[-\pi, \pi]$ but also on $[a, a+2\pi]$ for some real number a, as in (3).

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{\infty} \left(a_k \cos kx + b_k \sin kx \right)$$
(1)

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos nx \, dx, n = 0, 1, 2, \dots$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin nx \, dx, n = 0, 1, 2, \dots$$
 (2)

$$a_n = \frac{1}{\pi} \int_a^{a+2\pi} f(x) \cos nx \, dx, n = 0, 1, 2, \dots$$

$$b_n = \frac{1}{\pi} \int_a^{a+2\pi} f(x) \sin nx \, dx, n = 0, 1, 2, \dots$$
 (3)

In many real world audio processing situations, when f(x) is represented by a finite audio signal, it is unknown whether the series on the right side of (1) converges and actually equals the sum of f(x). Therefore, only the approximate equality in (4) can be used. Tolstov's method defines constituent harmonics in f(x) by (5).

$$f(x) \approx \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos nx + b_n \sin nx \right)$$
(4)

$$h_n(x) = a_n \cos nx + b_n \sin nx \tag{5}$$

The proposed algorithm for detecting A440 piano notes in beehive audio samples is based on Tolstov's method and implemented in Java. An A440 key K is represented as a harmonic defined in (5). Since n in (5) is a non-negative integer and note frequencies are real numbers, K is defined in (6) as a set of harmonics where f is K's A440 frequency in Hz and b is a small positive integer that defines a band of integer frequencies centered around K's real frequency. For example, the A440 frequency of A0 is 27.56 Hz. Thus, a set of six integer harmonics, from $h_{25}(x)$ to $h_{30}(x)$, can be defined to identify the presence of A0 in a signal, as in (7).

The presence of a key in a signal is done by thresholding the absolute values of the Fourier coefficients or harmonic magnitudes. Thus, a key is present in a signal if the set in (6) is not empty for a given value of b and a given threshold.

$$H_f(x) = \{h_n(x) | n \in [\lfloor f - b \rfloor, \lceil f + b \rceil]\}, b \in N^+$$
(6)

$$H_{27.56}(x) = \{ h_n(x) | n \in [25, 30] \}, b = 2$$
(7)

A Java class in the current implementation of the algorithm takes wav files and generates LOGO music files

for the LOGO Music Player (LMP) (www.terrapinlogo.com). Fig. 5 shows several lines from an LMP file generated from a wav file. PLAY is a LOGO function. The remainder of each line consists of the arguments to this function in square brackets. A single pair of brackets indicates a one-note chord; double brackets indicate a chord consisting of multiple resulting notes.

PLAY	[F7]												
PLAY	[[AO	A0#	в0	11									
PLAY	[[A1	A1#	в1	C1	C1#	D1	F1	F1#	G1	11			
PLAY	[C2]												
PLAY	[F7]												
PLAY	[[AO	A0#	11										
PLAY	[[AO	A0#	в0	11									
PLAY	[A0]												
PLAY	[[AO	A0#	в0	11									
PLAY	[[A1	A1#	в1	C1	C1#	D1	D1	‡ E1	F1	F1#	G1	G1#]]
PLAY	[[C2	C2#	D2	D2	# E2	G2 ;	#]]						
PLAY	[F7]												
PLAY	[B0]												
PLAY	[[C1	C1#	11										
PLAY	[A0]												
PLAY	[A0]												
PLAY	[A0]												
PLAY	[[AO	A0#	11										
PLAY	[[AO	A0#	в0]]									
Г	- 5 T	MD :				A 1.	440		1.4	4 1		C1	

Fig. 5. LMP instructions with A440 notes detected in wav files.





To visualize beehive music, the audio files generated by the LMP can be converted into musical scores. Fig. 6 gives one such score obtained with ScoreCloud Studio (http://scorecloud.com) from an LMP file converted to midi.

V. AUDIO DATA ANALYSIS

The audio digitization algorithm described in Section IV was applied to the buzzing signals collected by a BeePi device at the USU Organic Farm from 22:00 on July 4th, 2015 to 00:00 on July 7th, 2015. Each signal was saved as a 30-second wav file. A total of 152 wav files were collected in this period, which amounted to a total of 3421.52 MB of wav data. The digitization algorithm was applied to these data on a PC running Ubuntu 12.04 LTS.

The A440 piano keys were mapped to integers from 1 to 88 so that A0 was mapped to 1 and C8 to 88. In Fig. 7, these key numbers correspond to the values on the X axes. Each 24-hour period was split into 6 non-overlapping hour

ISBN: 978-988-19253-8-1 ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online) intervals: [0, 4], [5, 8], [9, 12], [13, 16], [17, 20], [21, 23]. The frequency counts of all notes detected at a frequency of 44100 Hz were computed for each interval. The detected spectrum contained only the first four octaves with the lowest detected note being A0 (1) and the highest F4# (5).



The buzzing frequencies exhibited a cyclic pattern during a 24-hour period. From 0:00 to 4:00 (see Fig. 7, top), the note range ran from A0 (1) to D1# (7), with the two most frequent notes being D1 (6) and C1# (5). From 5:00 to 8:00, the note range widened from A0 all the way up to F4#, with the two most frequent notes being D1# and E1.

From 9:00 to 12:00 (see Fig. 7, middle), the note range slightly narrowed to run from A0# (2) to C4# (41). The note frequency range had two pronounced sub-ranges. The first sub-range ran from A0# (2) to A1 (13), with the two most frequent notes being D1# (7) and E1 (8). The second sub-range ran from D2# (19) to D3 (30), with the two most frequent notes being F2# (22) and C3# (29). Compared to the interval from 0:00 to 4:00 (see Fig. 7, top), the frequency counts in the interval from 9:00 to 12:00 had significantly higher frequency counts, which may indicate that the bees were buzzing more actively.

From 13:00 to 16:00, the range narrowed to run from A0 (1) to F2 (21), with the two most frequent notes being C1 (4) and C1# (5). From 17:00 to 20:00, the range ran from A0 (1) to A4# (50), with the two most frequent notes being C3 (28) and C3# (29). From 21:00 to 23:00 (see Fig. 7, bottom), the range narrowed to run from A0 (1) to F3# (34) with the two most frequent notes being C1 (4) and C1# (5).

VI. ESTIMATING FORAGER TRAFFIC FROM IMAGES

Beekeepers use visual estimates of forager traffic to evaluate the health of a bee colony. In electronic beehive monitoring, computer vision can be used to estimate the amount of forager traffic from captured images. A sample image is shown in Fig. 8. The current forager traffic estimation algorithm is implemented in Java using the OpenCV 2 image processing library (www.opencv.org).

Since the camera's position is fixed, there is no need to process the entire image. The image processing starts by cropping a rectangular region of interest (ROI) where the landing pad is likely to be. The ROI is made wider and longer than the landing pad, because the camera swings slightly up, down and sideways in stronger winds, which causes the pad to shift up, down, left, or right.



Fig. 8. Image captured from the BeePi camera.

The ROI is brightened when its intensity level is below a threshold. Image brightening provides for more accurate bee identification in darker images (e.g., third image from the top in Fig. 9) captured on rainy or cloudy days. The actual landing pad is detected in the ROI with color histograms, because all landing pads in the hives, where the BeePi devices were deployed, have a distinct green color, as shown in Fig. 9. In Fig. 9, the two color images are the landing pads cropped from the ROIs.

The first bee identification algorithm implemented in BeePi is based on a contour detection algorithm in OpenCV 2. The algorithm takes a binarized image and returned a list of contours, each of which is a connected component of pixels. The contours with fewer than 30 or more than 50 pixels are removed. These parameters can be adjusted if necessary. The number of found contours is an estimate of the number of bees on the pad. In the two color images in Fig. 9, the red lines correspond to the contours of detected individual bees. The second bee identification algorithm is based on binary pixel separation of the cropped landing pad. Each pixel is inspected for the presence of the green color. If the presence of the green color exceeds a threshold, the pixel is labeled as a pad pixel. Otherwise, the pixel is labeled as a bee pixel. The second and fourth images in Fig. 9 show the white pad pixels and the black bee pixels. An estimate of the number of bees on the landing pad is obtained by dividing the number of detected bee pixels by 30, which is the average number of pixels in an individual bee.



Fig. 9. Bee detection at beehive's entrance.

TABLE I											
Computer vision vs human evaluation.											
Images	Bee Counts		М	ean	S	ACC					
270	CV	HM	CV	HM	CV	HM	72.07				
578	1582	2165	4.2	5.7	6.5	6.5	/5.07				

In the first experiment, to compare the accuracy of the two algorithms, 135 images of landing pads with bees on them were selected. The number of bees in each image was counted by a human observer. The total number of counted bees was 1204. Two algorithms were evaluated on all images. The contour detection algorithm accurately identified 650 bees of 1204 (54%). The background separation algorithm found 871 bees out of 1204 (71%).

In the second experiment, the pixel separation algorithm was compared with human evaluation on another sample of 378 images. The results are tabulated in Table I. The CV columns give the statistics for the pixel separation algorithm. The HM columns give the statistics of the two human evaluators who counted the actual numbers of bees in each of the 378 images. The first human evaluator processed 200 images. The second human evaluator processed the remaining 178 images. The human evaluators counted 2156 bees in 378 images. Human evaluation was used as the ground truth.

Of the 2156 bees found by the human evaluators, the pixel separation algorithm found 1582 bees in the same images. The overall accuracy, given in the ACC column, was computed as the ratio of the bees found by the algorithm and the bees found by the human evaluators. The accuracy came out to be 73%.

The Mean column in Table I shows the mean numbers of bees identified by the algorithm and the human evaluators in the images. These numbers indicate that the computer vision algorithm, on average, identifies fewer bees than the human evaluators in each image. The standard deviation (STD) column shows that standard deviations of the computer vision algorithm and the human evaluation are the same, which indicates that the algorithm is consistent.

The subsequent analysis revealed two main causes of

error for the pixel separation algorithm. The first cause was the wrong identification of the landing pad in bright images when the algorithm cropped not only the landing pad with bees but also chunks of grass. Some of the grass blades were erroneously counted as bees. The other cause of error was really dark images where the algorithm found smaller numbers of bees even after the images were brightened. Based on the experiments and observations, the pixel separation algorithm is proposed as a candidate method for estimating forager traffic levels from images.

VII. CONCLUSION

An electronic beehive monitoring device, called BeePi, was presented. Four BeePi devices were assembled and deployed in beehives with live bees over extended periods of time in different weather conditions. The field deployments demonstrate that it is feasible to use solar power in electronic beehive monitoring.

The algorithms were presented for digitizing bee buzzing signals into A440 piano note sequences and for estimating forager traffic levels from images. The algorithm for digitizing buzzing signals converts the wav signals into timestamped sequences of A440 piano notes. The detected note spectrum contained the first four octaves. The buzzing frequencies exhibited a cyclic pattern during a 24-hour period from the first octave all the way to the fourth octave and back.

Two computer vision algorithms were implemented and tested. The first method, based on a contour detection algorithm, takes a binarized image and estimates the bee counts as the number of detected contours containing between 30 and 50 pixels. The second algorithm is based on the binary pixel separation of the cropped landing pad into pad pixels and bee pixels. An estimate of the number of bees on the landing pad is obtained by dividing the number of the bee pixels by 30, which is the average number of pixels in an individual bee.

The pixel separation algorithm performed better than the contour detection algorithm on a sample of 135 images. The pixel separation algorithm was compared to the human evaluation on another sample of 378 images with an observed accuracy of 73%. Two main causes of error were individual grass blades detected as bees in bright images and dark images where some bees were not recognized.

ACKNOWLEDGMENT

The first author, Vladimir Kulyukin, expresses his gratitude to Neil Hattes who jumpstarted this project by donating a raspberry pi computer, a mother board, a temperature sensor, and a monitor. All data collection software was originally developed and tested on this equipment. The second author, Myles Putnam, and the third author, Sai Kiran Reka, contributed to this project pro bono. All bee packages, bee hives, and beekeeping equipment used in this study were personally funded by Kulyukin.

REFERENCES

 B. Walsh. "A World without bees," *Time*, pp. 26-31, August 19, 2013.
 M. T. Sanford. "2nd international workshop on hive and bee monitoring," *American Bee Journal*, December 2014, pp. 1351-1353.

- [3] Rev. L. L. Langstroth. Langstroth on the Hive and the Honey Bee: A Bee Keeper's Manual. Dodo Press: UK, 2008; orig. published in 1853.
- [4] B. N. Gates. "The temperature of the bee colony," United States Department of Agriculture, Dept. Bull. No. 96, 1914.
- [5] M.E.A. McNeil. "Electronic Beehive Monitoring," American Bee Journal, August 2015, pp. 875 - 879.
- [6] M. Bencsik, J. Bencsik, M. Baxter, A. Lucian, J. Romieu, and M. Millet. "Identification of the honey bee swarming process by analyzing the time course of hive vibrations," *Computers and Electronics in Agriculture*, vol. 76, pp. 44-50, 2011.
- [7] W. Blomstedt. "Technology V: Understanding the buzz with arnia." *American Bee Journal*, October 2014, pp. 1101 1104.
- [8] S. Ferrari, M. Silvab, M. Guarinoa, D. Berckmans. "Monitoring of swarming sounds in bee hives for early detection of the swarming period," *Computers and Electronics in Agriculture*. vol. 64, pp. 72 -77, 2008.
- [9] J. Rangel and T D. Seeley. "The signals initiating the mass exodus of a honeybee swarm from its nest," *Animal Behavior*, vol. 76, pp. 1943 -1952, 2008.
- [10] W.G. Meikle and N. Holst. "Application of continuous monitoring of honey bee colonies," *Apidologie*, vol. 46, pp. 10-22, 2015.
- [11] GNU Octave. https://www.gnu.org/software/octave.
- [12] G.P. Tolstov. Fourier Series. Dover Publications: New York, 1962.