Real-time ENF monitoring with a Raspberry Pi 3 Model B

Antonios Lykourinas and Athanassios Skodras
Department of Electrical and Computer Engineering
University of Patras
Patras, Greece
alykourinas@upnet.gr - skodras@upatras.gr

***Abstract*—Electrical Network Frequency (ENF), the instantaneous fluctuation of the power grid's frequency around its nominal frequency due to the imbalance between energy production and consumption, was initially used in forensics applications and since then it has been adopted into a wide range of applications. In this report, the development of a real-time ENF monitoring system is described. The whole system is based on a Raspberry Pi Model B computer, which performs the filtering of the digital signal, the calculation of the ENF values and finally the storage of the frequency values in real-time. Signal capturing is achieved by means of a sensing circuit that we have also designed and constructed. The performance of the system has been compared to existing solutions, thus validating its measuring accuracy.**

I. Introduction

The Electrical Network Frequency (ENF) refers to the instantaneous frequency of a power distribution network at a given time, The frequency of a power distribution network fluctuates in time around its nominal frequency. This nominal frequency depends on the geographical location of the distribution network. Wide area distribution networks operate at 60 Hz in the U.S. and at 50 Hz almost everywhere else in the world. The fluctuation around a nominal frequency is caused by the instantaneous imbalance between power generation and consumption [1].

ENF is intrinsically embedded in a lot of multimedia recordings. Initially, ENF's presence was detected in audio recordings obtained from devices connected directly to the power grid. However, further research has revealed that ENF traces can be present in audio recordings made by battery powered devices [2], where it has been proven that the main source of the traces is the audible hum produced by mains powered devices in close proximity [3]. ENF's presence has also been discovered in surveillance footage. The imaging sensor of a camera produces an image based on the intensity of the photons it receives. As indoor light sources operate at twice the nominal grid's frequency, the photons they emit contain ENF traces. These ENF traces are captured by the imaging sensors in form of illumination frequency [4].

A. ENF Extraction

Several methods have been introduced for ENF signal estimation, including zero crossings [1], conventional Short-Time Fourier Transform [5], spectrum combining [6] and Hilbert-Based [7] approaches to more advanced ones such as maximum likelihood estimation via a multi-harmonic model [8]. The most reliable method to collect strong ENF traces is by directly monitoring the power grid voltage thus reference databases contain ENF estimations extracted directly from processed power grid voltage signals. ENF extraction from multimedia recordings can be a challenging task.

Although it is widely accepted that audio recordings captured by devices directly connected to the power grid will contain ENF traces, this may not always be the case for battery-powered devices. Their ENF capture can be affected by several environmental factors and device-related scenarios [9]. It has also been proven that different microphones are able to capture ENF traces with higher SNR around different nominal frequency harmonics [9]. Several extraction methods can be used for extracting the ENF from audio recordings. In [10], authors used both a conventional short-time Fourier Transform for extracting the ENF from the camera's audio feed at the edge and a spectrum combining method for more robust ENF estimation at the Fog nodes.

In video recordings, ENF can be estimated by utilizing the aliased illumination frequency. Due to the older establishment of the PAL and NTCS video standards, cameras capture video at 23.98/29.97 frames per second [4]. According to Nyquist Theorem, the low sampling rate will cause the illumination frequency (100Hz/120Hz) to suffer from aliasing. Based on the imaging sensor, CCD or CMOS, different estimation methods have been developed due to the different capturing mechanisms that they use. Experiments with both sensors on ENF extraction have shown that CMOS sensors outperform CCD sensors by providing more accurate estimations [11].

B. ENF Applications

In [12], the author was asked to validate the authenticity of audio recordings, containing conversations between two speakers, which were brought to court as evidence. One of the speakers claimed that the audio recordings presented in court had been tampered. The recordings were captured with a computer and ENF traces were intrinsically embedded in the audio recordings due to the computer's direct connection to the power grid. These ENF traces were extracted and compared visually with recordings from a reference database. During the analysis, major discontinuities of the ENF were discovered, which supported the initial claims of tampering. This accomplishment led to the proposal of the ENF criterion, as a forensic technique for evaluating the authenticity of digital audio recordings [12]. Since then, the application spectrum of ENF signals has expanded. They have also been adopted for multimedia synchronization [13] and, more recently, for

preventing Frame Duplication Attacks at the edge [10], [14], [15].

C. ENF similarity measurement

For extensive use of the ENF criterion, an automated search routine was required in order to be able to locate the best matches between the analyzed multimedia recordings and a reference database [1]. Initially, the mean square error was used but due to the observation that a slight vertical offset could occur between ENF signals captured by different capturing devices [2], the need for a different comparison metric emerged. This metric was the correlation coefficient [1].

The correlation coefficient is associated with the equivalence of the patterns and achieves that by removing the DC component from both signals. The Correlation coefficient is widely adapted in ENF applications as it has been proved the most trusted metric in matching procedures.

D. Need for real-time ENF estimation

The adoption of ENF into a wide range of applications and the availability of estimation techniques has encouraged the development of ENF monitoring systems. Our first ENF monitoring system was developed in 2016 [16]. An improved version was developed five years later, which was based on a Raspberry Pi 3 Model B [17]. This last system has further been improved by designing a new signal capturing and conditioning circuit and new software enabling real-time visualisation of the ENF signal. Also, the improved system updates our Cloud database automatically.

The present technical report is organised as follows. In Section II, a brief summary of the existing systems is presented. In Section III, identified problems are discussed and solutions are proposed. In Section IV, we verify the validity of the system's results by comparing our estimations with estimations from other systems operating on the same power grid network; we also introduce some final minor adjustments allowing for a more robust ENF estimation. Finally, in Section V, conclusions are drawn followed by a brief discussion of the future work.

II. Existing ENF monitoring Systems

A. A power grid monitoring application

In 2016, a system was developed by a group of undergraduate students of the Department of Electrical and Computer Engineering of the University of Patras, Greece, with the aim to record the power signal directly from the power grid with the use of a computer [16]. They used an external circuit for sensing the power grid voltage, whose output was fed into a sound card through a 3.5 mm jack. The circuit included a transformer, a fuse for safety reasons followed by a voltage divider, and two passive filters; a low pass filter and a high pass filter. The transformer and voltage divider downgraded the voltage to levels appropriate for sensing, the high pass filter eliminated the DC component from the power grid signal, and the low pass filter served as an anti-aliasing filter, allowing sampling of the power grid voltage without any aliasing effects.

The signal was sampled at a sample rate of 1000Hz with the help of the PyAudio python module [18]. Each sample was represented by a 16-bit integer. After a normalization process the resulting signal was saved as a .wav file with the help of the wave python library [19]. The application monitored continuously the power grid for an hour. Then, the monitoring process was interrupted for the purpose of storing the recording, resulting in a small time interval between successive one-hour recordings.

A recording of 1 hour occupied approximately 7.20 MB (7.20 MB) (1000 samples/sec $\times$ 3600 sec $\times$ 2 bytes/sample) of disk space, while the ENF extraction had to be performed separately. At midnight, when the application detected that the day had changed, it compressed all .wav files containing all day's one-hour recordings into a zip archive. When the archiving process was over, the zip archive was automatically uploaded to the cloud.

B. A Raspberry Pi 3 Model B solution

In [17], a series of problems of the previous ENF monitoring application were identified, including the use of a computer for performing the relatively simple task of monitoring the power grid voltage, as well as the use of a single thread for both monitoring and archiving the one hour long .wav recordings and finally the direct storage of the .wav files without any pre-processing which leads to redundant data being stored.

In order to overcome the identified setbacks, the authors in [17] used a Raspberry Pi 3 Model B device, a handheld single-board computer equipped with an ARM processor and 1 GB of RAM, as the developing platform for the application. The monitoring and archiving processes were assigned to separate threads in order to achieve true continuous monitoring of the power grid. Furthermore, by extraction of the interested features prior to storage, the authors managed to decrease the size of the data resulting in redundant free, smaller files.

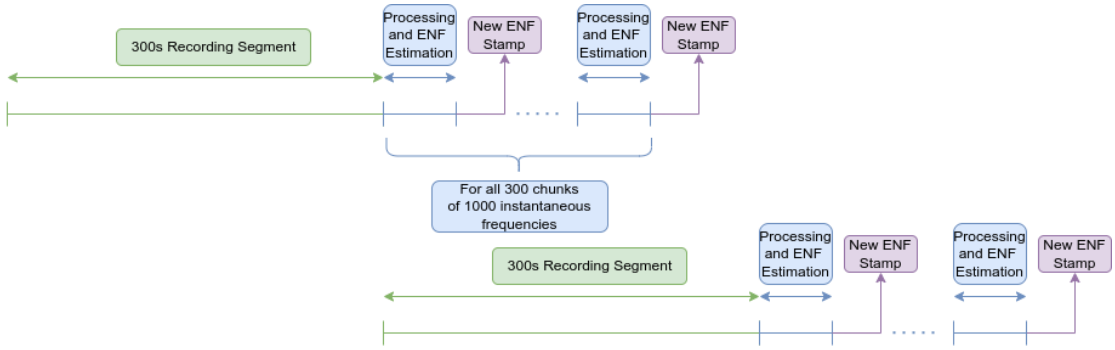
Additionally, new features were introduced and a different ENF extraction mechanism was used. The PyDrive [20] module of python was drafted for uploading the archived files into a Google Drive folder and by using the Hilbert-Based approach for estimating the ENF, the authors managed to achieve identical estimations with the initial application which used the zero crossings method [17].

III. Improvements on the Raspberry Pi system

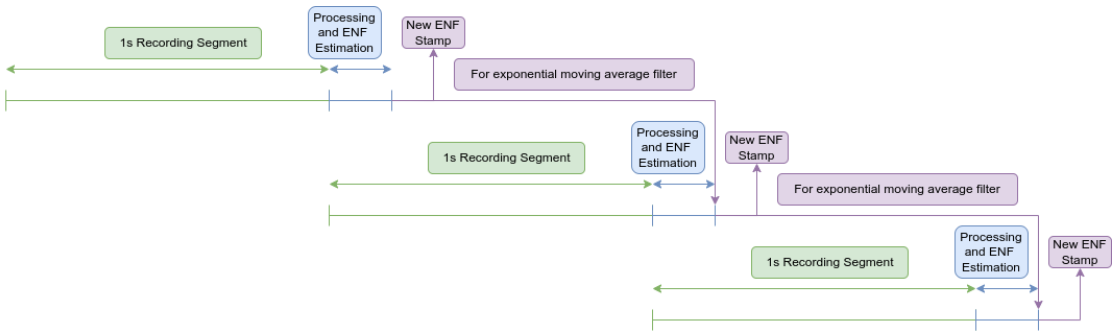
Although the authors in [17] provided solutions that managed to solve the problems that were identified, their system could be further improved.

A. Database creation

In the [17] system, a 300 second recording segment is processed as a single entity, then the ENF is estimated for each second within it. A timestamp is assigned to each ENF estimation based on the recording's starting time. The ENF stamp generated for each second is a string which contains an ENF estimation and its corresponding timestamp and it is formatted as follows, "yyyy/mm/dd HH:MM:SS frequency". These stamps are stored in a .csv file for every hour, and at the end of the day all hourly .csv files are compressed and uploaded to a Google drive folder.



(a) Old Application



(b) New Application

Fig. 1. Highlighting the differences between the two ENF extracting procedures.

In the existing system, all ENF estimations are available approximately 2 minutes after the 300 second recording segment is captured, which is the average time that the estimating thread needs to provide ENF estimations for every second within the segment. Although the ENF estimations are available after a short amount of time, the data generated can be accessed only after the current day is over, and the data become available online for download [21].

In order to overcome this problem and enable almost real-time manipulation of the data, a database was build. A MySQL database [22] was build with a table containing an id column, an ENF estimation column and a date-time column. With the use of mysql.connector module of python [23] the Raspberry Pi would now be capable of updating the database table each time a new ENF stamp was generated through a database handler.

A database handler library was also designed to simplify common interactions and queries performed on the database table through the application. For example, there is no need to keep all previous day's ENF estimations since they can be accessed from the google drive folder (we keep an amount of estimations in order to provide statistical data about ENF). Every time the application is restarted, a great margin of last days estimations could be discarded. In order to achieve this specific goal, a large set of operations and queries on the database table must be executed but now with the database handler this can be accomplished in a single line of code. The execution of frequently performed actions on the database table such as retrieve estimation by id or get latest estimation (used in the Exponential Moving Average Filter) is know simplified.

B. Delayed ENF estimation problem

In the existing implementation, the system captures the power grid voltage in segments of 300 seconds. The segments are later processed in a separate thread from the main one and an ENF estimation is produced for every second within the 300 second segment. The recording segment's duration is impractical for real-time ENF estimation. The final sampling rate of the signal is 1000 Hz. By utilizing the Hilbert-based approach, the application generates 300 thousand instantaneous frequencies (300 sec $\times$ 1000 samples/sec).

As reported in [17], based on empirical evidence, the first 20 instantaneous frequency samples are prone to large spikes and therefore are discarded. The remaining instantaneous frequency samples are then averaged per 1000 samples, producing a mean value that is processed by an exponential average filter with a smoothing factor of 0.1, resulting in the corresponding ENF estimation. The filter uses only the previous filtered value as a reference. This process is repeated until all instantaneous frequencies have been processed. Finally, the correct timestamp is assigned to each ENF estimation and their corresponding ENF stamps are generated.

The problem that we identified was that the 300 thousand instantaneous frequencies that were calculated for each segment, were processed independently into chunks of 1000 samples, indicating that the 300s recording duration of the segments was aimlessly selected for the ENF estimations. Each chunk of 1000 instantaneous frequencies is independent from each other and only the ENF estimation of the previous chunk is required for the next chunks estimation, as seen in Fig. 1(a).

After modifying the segment duration to 1 second the

whole power grid voltage processing and ENF estimation procedure was timed and the maximum execution time obtained was approximately 300ms. For every new segment, the mean value of the 1000 produced instantaneous frequency samples is calculated and the latest ENF estimation is requested with the database handler for the application of the exponential average filter. Finally, the correct timestamp is assigned to the ENF estimation and then the ENF stamp is generated and uploaded to the MySQL database. With these modifications the application has now become suitable for real-time applications. The ENF table is now updated every second.

C. Misuse of threads

Although separate threads are used for the archiving and estimation processes and are executed independently of the main thread, the archiving process is currently placed inside the main loop of the application. This can be improved by starting an archiving thread only once at the beginning of each day when the application is restarted, before the main loop begins. By starting an archiving thread at the start of each recording segment we aimlessly consume computational resources of the system, as no archiving is required until the application is restarted.

D. Uploading issues

An archived file containing all previous day's recordings is uploaded to a Google Drive folder every day. This was made possible by using PyDrive module of python. PyDrive is a client library for Google's API service. It simplifies many common Google Drive API tasks such us uploading, downloading and sharing files and folders.

Google Drive API Service relies on OAuth 2.0 for user authorization [24]. To use the API service, an OAuth 2.0 Client ID for the user of the desktop application must be created. Once the Client ID is created and downloaded in .json format, the python application will request from the user to login in order to use the Google's API services. After the user is logged in, a .json file containing an access token is generated. Access token is a string that Oauth2 clients use to make requests to the resource servers.

The main problem with the existing system was that the access token expired after 3600 seconds [25], resulting in requesting from the user to manually enter his/her credentials every time a new file was ready to be uploaded. To address this issue, the OAuth 2.0 standard offers refresh tokens. Refresh token is a string that the OAuth client can use to get new access tokens without user's re-authentication. Refresh tokens don't usually expire, but can be expired for a series of different reasons, according to information Google provided to developers using their API Services [25]. Additionally, the application status must be changed from "Testing" to "Published" otherwise, according to Google [25], refresh tokens only last 7 days.

The Authorization process was modified and through authentication a refresh token was now received. The application now can successfully upload the new archived .csv recordings without user's intervention.

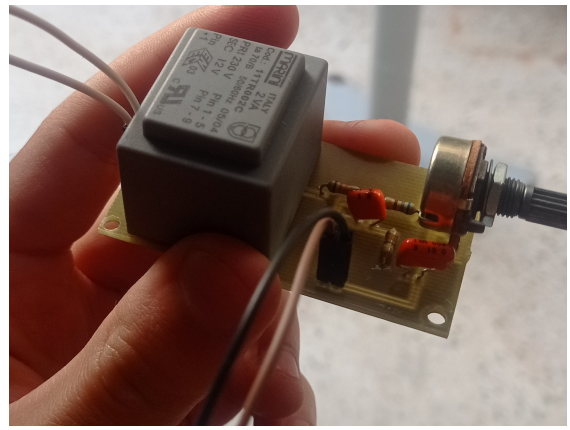


Fig. 2. The power grid voltage sensing circuit

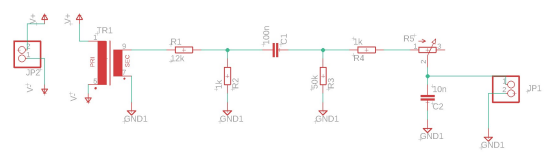


Fig. 3. The schematic of the power grid voltage sensing circuit

E. Making the code more understandable

In [17], although the authors did great work with providing information about each function and thread they declared, the whole application was written as a single source file, making it unreadable and difficult to understand the program flow. Therefore, the application was separated in different modules each one corresponding to a different aspect of the system. Recording functions, signal processing functions, intrinsic logging mechanisms, and the database-handler were all developed as different modules. The program flow could be separated into different levels of abstraction thus making it easier to identify bugs and to read the source code.

F. Updating the log mechanisms

The application logs important information such as the starting times as well as successful and unsuccessful file uploads. However, while both successful and unsuccessful uploads are tracked, the corresponding archived files of those uploads are not. As a consequence, archived folders that weren't uploaded will require the user's manual intervention for this to be achieved.

In order to overcome this issue, a dedicated module was created to handle all the intrinsic logging tasks. This new module tracks the status of every file upload, including both successful and unsuccessful attempts, and maintains a list of all archived files along with their corresponding upload status.

At every restart of the application, a separate archiving thread is created. The archive thread reads the list of all archived files along with their corresponding upload status and the uploading process repeats for every unsuccessful upload.

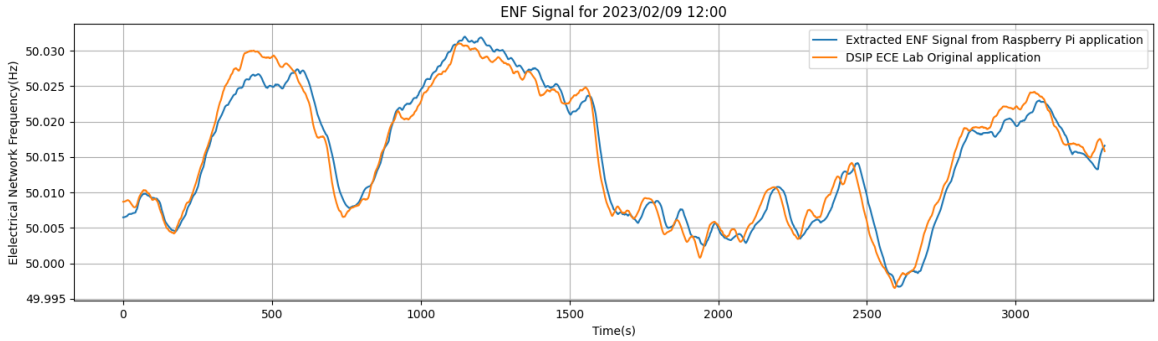


Fig. 4. ENF signals captured by DSIP ECE Lab's existing system and by the new Raspberry Pi system.

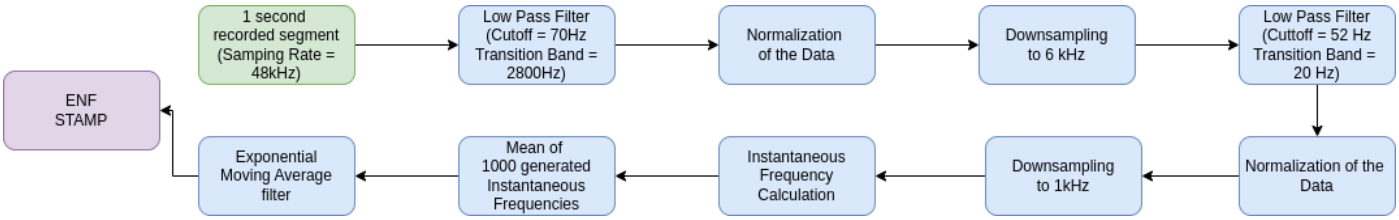


Fig. 5. The whole process of ENF estimation from the segment capture.

G. External circuit improvements

In the existing systems, the external sensing circuit was built on breadboard thus rendering the whole system bulky [16].

A smaller transformer with a lower secondary voltage could be used since the transformer is not driving any load and is only used for voltage sensing. In addition, the circuit draws about 5 mA of RMS alternating current so even small 1/4 watt resistors could be used. Although breadboard circuits are not necessarily fragile, they do not offer the convenience of a Printed Circuit Board (PCB). A Faraday cage is formed by shielding copper traces with metal cages on a printed circuit board resulting in better isolation from external electromagnetic interference. Furthermore, by using a PCB, we reduce the possibility of human error in the assembly process while also shortening its duration. In addition, PCB design software allows more efficient placing of the components under space restrictions resulting into smaller circuits.

A circuit was designed in Autodesk EAGLE [26] and a PCB was created for the reasons stated above. Figures 2 and 3 show the sensing device and the schematic, respectively. A smaller 2VA transformer with a 9V secondary voltage, 1/4 watt resistors, and ceramic capacitors were used. The dimensions of the PCB circuit are $6.5cm \times 3.5cm$. The sensing circuit and Raspberry Pi could now be placed inside a small case, enabling portable ENF estimation.

IV. Results

A. Application accuracy

The application has been running unsupervised for the last two months and all observed bugs have been fixed. The archived files are stored in a Google Drive folder [21]. The accuracy of the ENF estimations has been verified. The ENF estimations were compared with estimations coming from a system developed by a group from the Aristotle University of Thessaloniki [27], Greece, the DSIP ECE Lab's existing system and finally a device used by the Electric Power Systems Section of the ECE Department. A correlation coefficient of 0.985 was obtained between the new Raspberry Pi based ENF estimation system and that of the DSIP Lab's existing system. (Fig.4). An Exponential Moving Average filter with a smoothing factor of 0.01 was applied to both ENF signals.

B. Minor adjustments

When the duration of segments used for the ENF estimation was changed from 300 seconds to 1 second, the recorded ENF signals exhibited sudden spikes. As mentioned in [17], the 20 first instantaneous frequencies were discarded because of their high instability but the reason behind this instability was not revealed. The frequency instability issue is caused due to the application of the FIR filters. The output of an FIR filter exhibits a transient state whose length is equal to the length of the filter's coefficients and it reaches a steady state only when all coefficients are multiplied with real input signal samples [28]. As a result, a number of samples at the beginning of each processed recording segment are not properly filtered and their instantaneous frequencies, produced by the Hilbert-based approach, exhibit sudden frequency spikes.

In our application, we use the np.convolve numpy function [29] in order to apply the digital filters to the recording segments. The function's 'same' mode is selected, which produces an output signal with a length equal to the maximum length between the two input signals. This causes transient state effects both in the beginning and the end of the processed recording segment, resulting into sudden spikes also at the last instantaneous frequencies. By modifying the recording's segment duration we reached the the following conclusion. As the duration of the recording segment decreases, the calculated

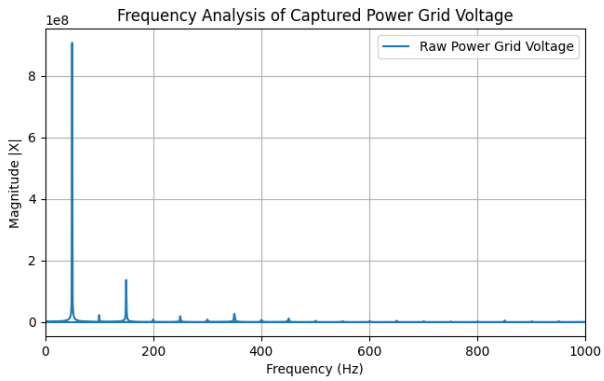


Fig. 6. Frequency response of the raw power grid signal.

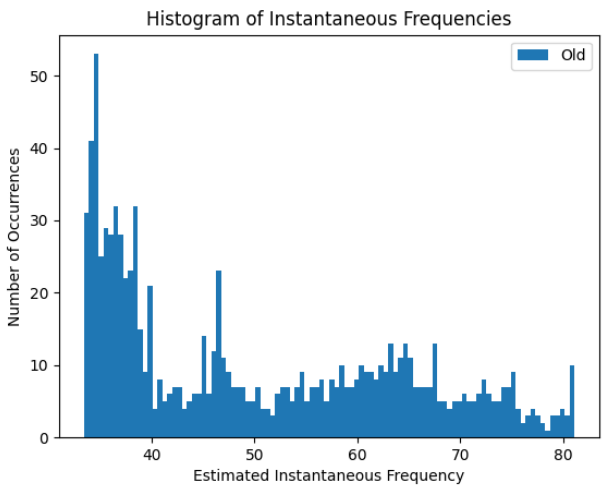


Fig. 7. Instantaneous frequencies obtained for a 1 second segment using the previous application.

instantaneous frequencies for this segment decrease, and the negative impact that instantaneous frequency spikes have on the final ENF estimation becomes more noticeable.

We also noticed that estimating the ENF with the Hilbert-Transform based method after applying the two digital filters, as depicted in Fig. 5, resulted into noisy ENF captures even when we discarded the first 100 and last 100 instantaneous frequencies. By examining a one second segment separately, we noticed that strong ENF traces were still present in harmonics of the nominal frequency because all frequencies below 500 Hz weren't properly filtered (Fig. 6).

Upon examining the histogram of 1000 instantaneous frequencies obtained from a 1 second recording segment, it was observed that the data exhibited a high degree of variance (Fig. 7).

Our initial assumption was that the presence of nominal frequency harmonics resulted in the high degree of variance of the instantaneous frequencies. As can be seen in Fig. 6, ENF traces were found in the frequency range of 0 to 500 Hz. A 30-tap low pass Remez filter was designed, which served as an anti-aliasing filter for down-sampling to 3 kHz. Then a 240-tap band pass Remez filter was designed in order to isolate the 50

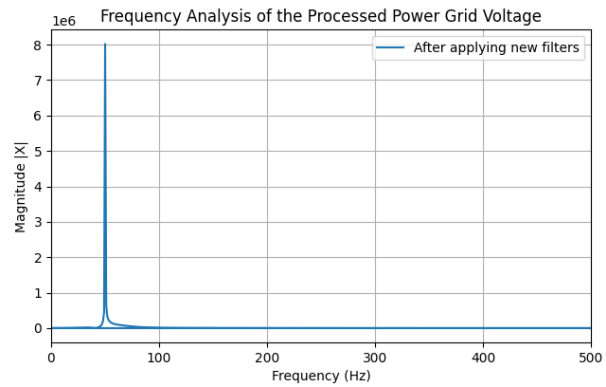


Fig. 8. Frequency response of the processed power grid signal after the new filters were applied.

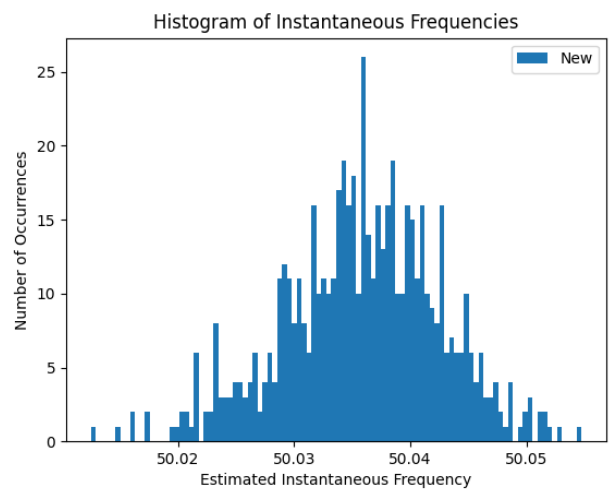


Fig. 9. Instantaneous frequencies obtained for a 1 second segment after the new filters were applied.

Hz frequency component. Finally, the Hilbert-based approach was used and the instantaneous frequency of each sample was obtained. Taking into consideration the transient effects caused by the application of the FIR filters, we chose to discard both the first 200 and last 200 instantaneous frequency samples. With these modifications the sudden frequency spike issue was solved allowing for more accurate ENF estimations (Fig. 8 and 9).

C. Frequency- Offset

As we can see in Fig. 10, a slight offset is observed between the existing system and the new one where the adjustments we described are incorporated. Identical offsets have been observed when we compared our ENF estimations with ENF estimations from different systems in the same power grid and they were first discovered in [2]. This offset doesn't concern us because it is neglected by the Correlation Coefficient [1]. We assume that filtering all the other nominal frequency harmonics caused the offset's appearance.

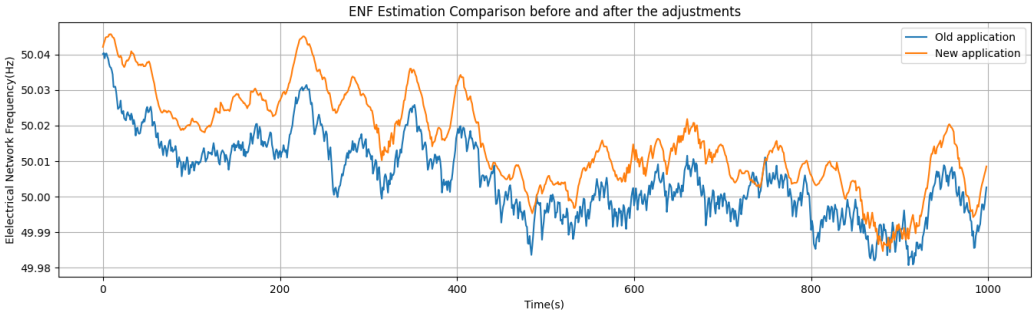


Fig. 10. ENF signals before and after the adjustments were made.

V. Conclusions

By improving the already existing Raspberry Pi system, we were able to capture and estimate the ENF in real-time. The addition of a database and a database-handler enabled real-time visualization of the ENF estimations. By retrieving a refresh token from the Google API service, files could now be automatically uploaded to Google drive without any user intervention. Creating a PCB circuit fitting in a case enabled portable estimation of the ENF while ensuring component safety and reducing electromagnetic interference. The misuses of threads were spotted and corrected. The log mechanisms of the application were updated. Monitoring app results during run-time led to slight modifications of the signal processing procedure of the captured power grid voltage. A lot of problems that resulted in noisy ENF estimations were identified and resolved. Finally, robust real-time estimation of the ENF was achieved.

Acknowledgment

The help of Dimitrios Tsipianitis (Electric Circuits and Systems Laboratory - Department of Electrical and Computer Engineering) in the construction of the PCB sensing circuit is gratefully acknowledged.

References

[1] M. Huijbregtse and Z. Geradts, "Using the ENF Criterion for Determining the Time of Recording of Short Digital Audio Recordings," in *Computational Forensics*, Z. J. M. H. Geradts, K. Y. Franke, and C. J. Veenman, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 116–124.

[2] M. Kajstura, A. Trawinska, and J. Hebenstreit, "Application of the Electrical Network Frequency (ENF) Criterion: A case of a digital recording," *Forensic Science International*, vol. 155, no. 2, pp. 165–171, 2005. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0379073804007662

[3] N. Fechner and M. Kirchner, "The Humming Hum: Background Noise as a Carrier of ENF Artifacts in Mobile Device Audio Recordings," in *2014 Eighth International Conference on IT Security Incident Management & IT Forensics*, 2014, pp. 3–13.

[4] R. Garg, A. L. Varna, A. Hajj-Ahmad, and M. Wu, ""seeing" ENF: Power-Signature-Based Timestamp for Digital Multimedia via Optical Sensing and Signal Processing," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 9, pp. 1417–1432, 2013.

[5] A. J. Cooper, "An automated approach to the Electric Network Frequency (ENF) criterion - Theory and practice," *International Journal of Speech, Language and the Law*, vol. 16, no. 2, p. 193–218, Apr. 2010. [Online]. Available: https://journal.equinoxpub.com/IJSLL/article/view/6099

[6] A. Hajj-Ahmad, R. Garg, and M. Wu, "Spectrum Combining for ENF Signal Estimation," *IEEE Signal Processing Letters*, vol. 20, no. 9, pp. 885–888, 2013.

[7] A. Triantafyllopoulos, I. Krillis, A. Foliadis, and A. Skodras, "A Hilbert-Based Aprroach to the ENF Extraction Problem," *IEICE Proceedings Series 24.A3-3*, 2016.

[8] D. Bykhovsky and A. Cohen, "Electrical network frequency (enf) maximum-likelihood estimation via a multitone harmonic model," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 5, p. 744–753, 2013.

[9] A. Hajj-Ahmad, C.-W. Wong, S. Gambino, Q. Zhu, M. Yu, and M. Wu, "Factors Affecting ENF Capture in Audio," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 277–288, 2019.

[10] D. Nagothu, Y. Chen, E. Blasch, A. Aved, and S. Zhu, "Detecting Malicious False Frame Injection Attacks on Surveillance Systems at the Edge Using Electrical Network Frequency Signals," *Sensors*, vol. 19, no. 11, 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/11/2424

[11] E. Baktalia, "Electric Network Frequency extraction from videos," Integrated Master's Thesis, University of Patras, Patras, Greece, 2023. [Online]. Available: https://nemertes.library.upatras.gr/items/3423e19a-8e4c-4107-a967-83c6c208e159

[12] C. Grigoras, "Digital Audio Recording Analysis: The Electric Network Frequency (ENF) criterion," *International Journal of Speech, Language and the Law*, vol. 12, no. 1, p. 63–76, 2005.

[13] H. Su, A. Hajj-Ahmad, M. Wu, and D. W. Oard, "Exploring the use of ENF for multimedia synchronization," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 4613–4617.

[14] D. Nagothu, Y. Chen, A. Aved, and E. Blasch, "Authenticating Video Feeds using Electric Network Frequency Estimation at the Edge," *EAI Endorsed Transactions on Security and Safety*, vol. 7, no. 24, 2 2021.

[15] A. Lykourinas and A. Skodras, "Video Surveillance Authentication: Real-time ENF Signal Hiding Technique at the Edge," in *Proceedings 2023 24th International Confererence on Digital Signal Processing*, 11-13 June 2023, Rhodes Island, Greece. [Online]. Available: https://ieeexplore.ieee.org/

[16] A. Triantafyllopoulos, A. Foliadis, G. Roustas, F. Athanasiou, and M. Papaioannou, "Exploring Power Signatures for Location Forensics of Media Recordings," *IEEE Signal Processing Cup 2016*, 2016.

[17] C. Frantzolas, "Electrical Network Frequency recording using a Raspberry Pi 3, Model B," Department of Electrical and Computer Engineering, University of Patras, Tech. Rep. DSIP/TR001, 2021. [Online]. Available: http://dsip.ece.upatras.gr/wp-content/uploads/2021/03/DSIP-TR001-10.03.2021.pdf

[18] "PyAudio." [Online]. Available: https://pypi.org/project/PyAudio/

[19] "Wave - read and write WAV files." [Online]. Available: https://docs.python.org/3/library/wave.html

[20] "PyDrive." [Online]. Available: https://pypi.org/project/PyDrive/

[21] "DSIP Lab (Digital Signal and Image Processing Laboratory): ENF database for Mainland Greece." [Online]. Available: https://dsip.ece.upatras.gr/projects/

[22] "MySQL Database." [Online]. Available: https://www.mysql.com/

[23] "MySQL Connector/Python Developer Guide." [Online]. Available: https://dev.mysql.com/doc/connector-python/en/

[24] "OAuth 2.0." [Online]. Available: https://oauth.net/2/

[25] "Using OAuth 2.0 to access google apis nbsp;—nbsp; authorization nbsp;—nbsp; google developers." [Online]. Available: https://developers.google.com/identity/protocols/oauth2

[26] "Eagle: PCB design and electrical schematic software," Aug 2022. [Online]. Available: https://www.autodesk.com/products/eagle/overview?term=1-YEARamp;tab=subscription

[27] G. Karantaidis and C. Kotropoulos, "ASSESSING SPECTRAL ESTIMATION METHODS FOR ELECTRIC NETWORK FREQUENCY EXTRACTION," in *Proceedings of the 22nd Pan-Hellenic Conference on Informatics (PCI 2018)*. New York, NY, USA: ACM, 2018, pp. 202–207.

[28] G. Moustakides, *Basic techniques for digital signal processing*. Filippou 91, 546 35 Thessaloniki: Tziola Publications, 2019.

[29] "The NumPy convolve function." [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.convolve.html