Measuring Spectrum Similarity in Distributed Radio Monitoring Systems

Roberto Calvo-Palomino[1,2], Domenico Giustiniano[1] and Vincent Lenders[3]

[1] IMDEA Networks Institute, Madrid, Spain,
{roberto.calvo,domenico.giustiniano}@imdea.org,
[2] Universidad Carlos III of Madrid, Spain
[3] Armasuisse, Thun, Switzerland.
vincent.lenders@armasuisse.ch

Abstract. The idea of distributed spectrum monitoring with RF front-ends of a few dollars is gaining attention to capture the real-time usage of the wireless spectrum at large geographical scale. Yet the limited hardware of these nodes hinders some of the applications that could be envisioned. In this work, we exploit the fact that, because of its affordable cost, massive deployments of spectrum sensors could be foreseen in the early future, where the radio signal of one wireless transmitter is received by multiple spectrum sensors in range and connected over the public Internet. We envision that nodes in this scenario may collaboratively take decisions about which portion of the spectrum to monitor or not. A key problem for collaborative decision is to identify the conditions where the nodes receive the same spectrum data. We take an initial step in this direction, presenting a collaborative system architecture, and investigating the challenges to correlate pre-processed data in the backend, with key insights in the trade-offs in the system design in terms of network bandwidth and type of over-the-air radio signals. Our results suggest that it is feasible to determinate in the backend if two sensors are reading the same analog/digital signal in the same frequency, only sampling during 200 milliseconds and sending just 1 Kbyte of data per sensor to the backend.

1 Introduction

Networked and distributed infrastructure using spectrum analyzers connected over the public Internet are emerging as an attractive alternative to traditional methods to monitor the spectrum usage based on expensive and specialized hardware of governmental agencies [1–3]. More recently, commoditized low-cost RF front-end are allowing to trade-off radio devices with high sensitivity, low-sweeping time and small frequency offset with low-cost spectrum monitoring hardware and software-defined signal processing in order to build a pervasive and affordable solution [4–7]. The main advantage of low-end distributed solutions is that they may allow to create a wide-scale and real-time spectrum monitoring system. A common assumption of most of concepts proposed in the literature is that they consider the different spectrum sensors as separate entities, each responsible to monitor the spectrum in a given area and with fully autonomous decisions [8].

However, in a crowded deployment, sensors may partially sense the same spectrum. We then envision a collaborative environment where a large number of sensing nodes work together in a coordinated manner. Only recently, some work has shown that collaboration among sensors is possible, and transmitting I/Q samples to the backend, data can be decoded in the backend with a distributed time-division scheduler [9]. The envisioned application in [9] requires a high network bandwidth load per device (in the order of a few or tens of Mb/s, depending on the number of sensors in range). Complementary to that work, we consider the goal of identify spectrum sensor nodes in coverage of the same transmitter. This problem requires to find a solution that i) takes fast decisions, so that nodes may quickly verify if they are in range of the same transmitter, and ii) needs low network bandwidth to minimize the load on the network. Applications of this knowledge could be several. For instance, collaboration to reach a common goal has the potential to reduce the time to monitor the spectrum using low-cost hardware. This could alleviate a main problem of current low-cost spectrum sensors deployments, that require between 40 and 70 seconds to sweep a spectrum of 1.7 GHz of bandwidth [7]. As a result of the frequency-dependent propagation characteristics, two nodes measuring the same or similar spectrum could then make a joint decision to avoid to both listening to the same portion of the spectrum, and instead spend more time on frequencies with little similarity with other nodes. Another exemplary application is the one of a sensor that finds out an anomaly in the signal. Here the node may ask the help of other sensors to confirm this anomaly.

In this work, we demonstrate and implement a system architecture able to collect the sensor's spectrum data and determine the similarity of the radio signal received at nearby locations by different sensing nodes. Our work provides the following key contributions:

- We overcome the limitation of CPU resources of embedded nodes, which dramatically reduce the time that a node can dedicate to gather spectrum samples, by proposing a synchronous sampling process in the sensors to determine the precise moment in time where multiple sensors can read the spectrum at the same time.
- We reduce the bandwidth used per each sensor by operating in the Fast Fourier Transform (FFT) domain and averaging the magnitude FFT on the sensors in order to send the minimum amount data needed to detect similarity to the backend.
- We evaluate and demonstrate a procedure to compute the spectrum correlation of the signals read by sensors, and apply it to analog and digital channels with spectrum resolution of 9.3 kHz.

We demonstrate that our system can reliably correlate real data spectrum (analog and digital signals) collected by different sensors sampling during 200 milliseconds and sending only 1 Kbyte of data per sensor to the backend.

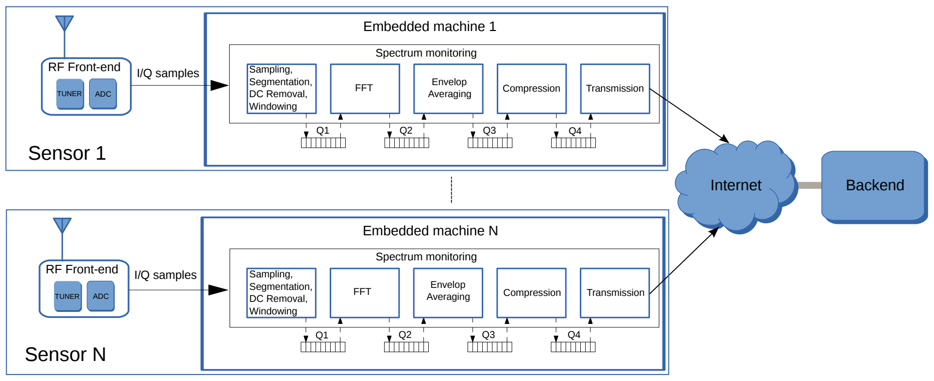


Fig. 1. High level architecture of each spectrum sensing node and overview of the overall system.

The remainder of this paper is organized as follows. In Section 2 we present the sensor platform. We then present the main research challenges in Section 3 and the system methodology for distributed spectrum correlation in Section 4. The evaluation of the proposed concepts is the focus of Section 5. We then present related work in Section 6 and finally draw the conclusion in Section 7.

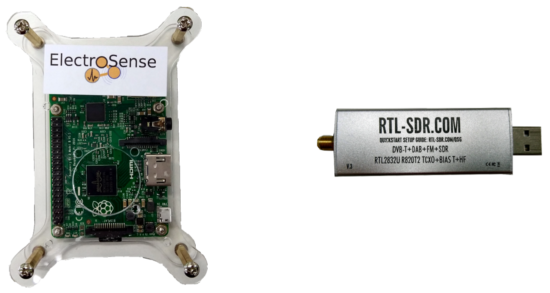
2 Platform for Collaborative Spectrum Sensing

This section describes the main concepts that feature our low-cost spectrum sensing platform for distributed and collaborative spectrum data collection. A high-level representation of the system is shown in Fig. 1.

2.1 RF data acquisition

Our architecture supports any RF front-end as long as it reports the raw I/Q samples over USB interface. To sample the spectrum, we rely on low-cost RTL-SDR USB dongles as RF interfaces, that support continuous tuning range between 24 MHz and 1766 MHz [7]. In this work we use the recent RTL-SDR *'Silver v3'*[4] device (shown in Fig. 2(b)) that has a nominal maximum frequency error of 1 part-per-million (PPM) and a temperature compensated crystal oscillator (TCXO). The TCXO allows to keep stable the frequency offset even with changing temperature environments. We use the C library *librtlsdr* from the OsmocomSDR project[5] to acquire the digital samples. To manage the spectrum samples, we rely on low-cost embedded machines. Figure 2(a) shows the RaspberryPi (RPi), that provides one CPU and GPU core[6]. For network connectivity, we rely on the on-board Ethernet.

[4] http://www.rtl-sdr.com/buy-rtl-sdr-dvb-t-dongles/
[5] http://osmocom.org
[6] In details, the RPi is model 3, with 1.2 GHz 64-bit quad-core and 1 GB of RAM



(a) Raspberry Pi (b) RTL-SDR device

Fig. 2. Hardware used.

2.2 Software-Defined Radio Sensor

The spectrum monitoring module of our sensors accesses the RF front-end to retrieve I/Q samples and sends magnitude FFT samples to the backend. It implements in software the necessary signal processing mechanisms for spectrum analysis such as sampling, segmentation, windowing, DC removal, FFT, envelope detection, etc in different software threads. A preliminary design and implementation of our sensor was presented in [7]. The FFT is computed in the GPU, since it is more efficient than the CPU for parallel signal processing[7].

The different threads are connected through queues, so that processing steps get decoupled from one another and in order to increase the efficiency in processing (e.g. blocks can be configured to process the queuing systems items at different granularities, i.e. one by one or in batches of several items). In order to prioritize the samples' reading over other threads, the main signal processing tasks of our architecture (Sampling, FFT, Envelope Detection, Compression and Transmission) are scheduled using default scheduler of Linux systems (Completely Fair Scheduler, CFS) with FIFO policy, which ensures that these tasks have a high priority in the system in terms of execution with respect to any other tasks in the system.

Each node of our system is able to compute magnitude FFT samples over a signal bandwidth of 2 MHz. (The node collects 2.4 MHz of spectrum, but we remove the edge of the spectrum where we observe that the frequency response is not flat.) Data are sensed, analyzed and compressed without any losses, with a default spectral resolution of 9.3 kHz. In the default configuration, an averaging factor of five is used in the data (i.e. 5 FFT blocks are used for averaging). In this configuration, uploading compressed data streams require a rate of 750 Kb/sec when continuously monitoring a fixed band of 2 MHz. As a result of the time spent to switch frequency, the uplink rates is reduced to 120 kb/s when the sensor is instructed by the backend to hop across the full wideband spectrum of

[7] For instance, we have experimentally measured that batches of 100 FFTs of 256 samples/bins each are computed in ≈ 10 us using the GPU of the RPi, almost four time less that using the CPU.

1742 MHz. The code implemented in the sensors is available as open source on github[8].

2.3 Backend

The backend is a server or set of servers with sufficient large storage and computation capabilities. Its main responsibility is to control the sensors and store, process and decode data received from the sensors, and perform data post-processing.

3 Challenges

Measuring the similarity between data spectra collected by independent sensors at different locations and connected to a backend over the Internet could allow us to explore intelligent collaborative decisions. For instance, the entire spectrum could be monitored faster. In another application, sensors may collaborate to detect anomalies if they learn that they are in range of the same transmitter in that specific frequency band. Also, sensors may decide to monitor a bandwidth larger than the one of the single sensor (with each node sampling on a 2 MHz band only partially overlapped with other sensors in range) and then jointly reconstruct the spectrum more efficiently in the backend.

In order to realize this vision, there are three fundamental challenges that need to be addressed with low-cost spectrum sensors:

- **Challenge 1: Time Synchronization.** Each node reads spectrum data, performs some pre-processing and sends it to the backend for post-processing. As nodes heavily relies on software-defined radios, delays are possible in the pre-processing in the sensors, which would adversely affect any attempt to compare data from different sensors. Therefore, it is essential to have synchronized information in time.
- **Challenge 2: Rapid decision making.** The system should take quick decisions about if a set of spectrum sensors can work collaboratively to achieve a common goal. The amount of analyzed data plays an important role to converge to a decision in the shortest possible time.
- **Challenge 3: Network bandwidth used.** The sensors send spectrum data to the backend where a decision is taken in order to allow the collaborative approach. The more data sent, the better the performance in the backend. Yet the greater is the bandwidth used. The system should find the right trade-off to provide enough context information to detect similarity in the spectrum using the minimum network bandwidth.

[8] It can be downloaded at https://github.com/electrosense/es-sensor.

4 Measuring Spectrum Similarity

In this section, we describe the methodology proposed to address the challenges presented in Section 3. The methodology is composed by three main concepts.

First, the sensors have a coarse synchronization in time using Network Time Protocol (NTP). The sensors can then read spectrum data with synchronization with millisecond precision over the public Internet, and better that one millisecond precision in local area networks (as measured in ad-hoc experiments). In our architecture, each sensor synchronizes its internal clock with NTP server every 60 seconds. A finer synchronization could be achieved using the time reference provided by an additional GPSDO module, as supposed by previous work in this area [9]. Yet, this may not be available in all the spectrum sensors, as they are run by users of the community, with they own hardware.

Second, each spectrum sensor activates the sampling thread for a fixed period, and then executes the other threads. This period is the same for each sensor. The total time to execute the entire set of operations should be such that the embedded machine can send spectrum data with the minimum delay (i.e. there are no other tasks pending).

Finally, in order to reduce the network bandwidth used, some operations need to be computed on the sensors as described in Section 2. Sensors should decide the parameters such as averaging over larger set of FFT blocks, if the bandwidth should be reduced further. This however comes at the cost of additional delay in the decision making. In the following section we describe this problem in details.

4.1 Spectrum Correlation

We perform signal correlation analysis in the backend to understand the similarity in the spectrum data collected by different sensors. Figure 3 shows the overview of the architecture that is responsible to compute the spectrum similarity (it is presented for two sensors, but it could be generalized to any number of sensors). First, the same center frequency and *sampling_time* is set in both sensors. During this sampling time the sensors continuously read the spectrum. After that, the raw data spectrum is split in N segments of *duty_cycle* long each one. For every segment, the FFT is computed according to the *fft_size* (which defines the number of bins). Each *fft_block* has the spectrum information of 2.0 MHz bandwidth channel. Therefore, the FFT blocks contain frequency information that is averaged using *avg_factor* blocks to produce M averaging blocks. These averaging blocks are sent to the backend for every sensor involved in the similarity spectrum evaluation. Tuning the different values of the system (*sampling_time*, *fft_size* and *avg_factor*) the amount of data sent to the backend (bandwidth used) can be modulated depending of the needs.

The backend computes the Pearson correlation coefficient using the averaged FFT blocks received from every sensor. The correlation is computed for every pair of averaged FFT blocks coming for different sensors. Then, the average is computed using the correlation output for every averaging block. The system

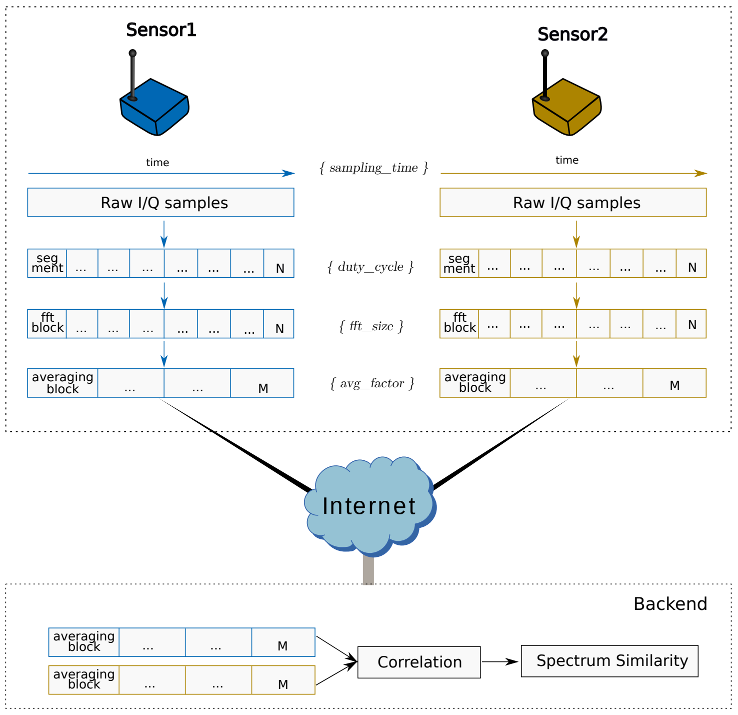


Fig. 3. Technique for computing spectrum similarity between two sensors.

evaluates this value to determine if the sensors are reading similar spectrum and if it is possible to enable the collaborative approach between them.

5 Evaluation

This section presents the experimental evaluation of our architecture. The aim is to determine the minimum network bandwidth and time needed to achieve a good signal correlation for analog and digital signals. For analog signals we choose FM and for digital signals we use Long-Term Evolution (LTE). The FM radio in Spain occupies the range from 87.5 MHz to 108 MHz. Each FM radio channel uses 200 kHz of bandwidth to transmit the signal. For LTE, we monitor three channels of 10 MHz bandwidth that are available in Spain at center frequencies equal to 796, 806 and 816 MHz.

Sampling Process' Synchronization. The synchronization between sensors is an important issue to achieve a good correlation between the signals. In our experiment, the sensors use Ethernet connection through Internet to synchronize their internal clock through NTP. This technology provides us an accuracy of 0.1 ms in a local network scenario and around a millisecond over Internet. The backend is responsible to define the precise moment when the sensors start to sample at the same time. This operation provides a coarse synchronization

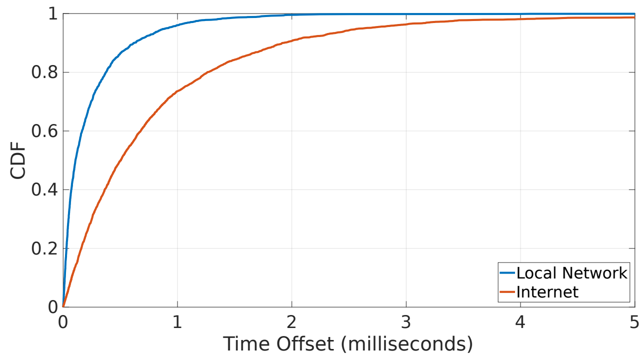
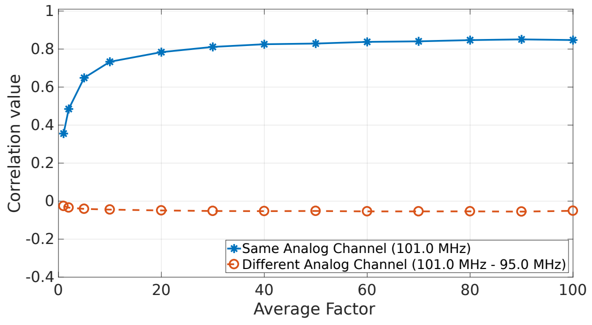


Fig. 4. Time offset between sensors in LAN and Internet.

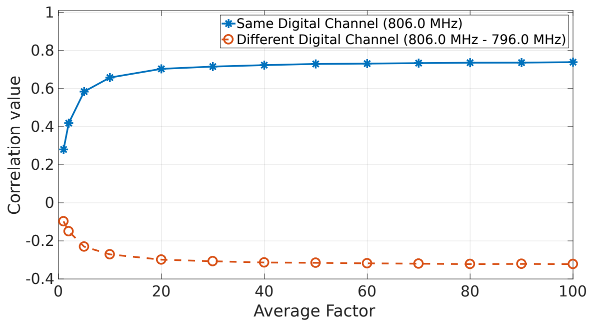
due to the delays added by the network and the operative system. Figure 4 shows the time offset between sensors when they are in the same local network (LAN) or connected through Internet. The 90% of the time the sensors have a time offset lower than 2 milliseconds when they are connected through Internet. In our solution we are interested to enable the collaboration among sensors connected through Internet (no just in the same local network). This provides a lower bound for the amount of data to be transmitted from the sensors for spectrum similarity analysis.

Spectrum Correlation. The spectrum correlation value is the metric used to determinate if the sensors are receiving a similar signal in the same frequency at the same time. The data provided by each sensor covers 2.0 MHz of bandwidth. The spectrum resolution in this bandwidth is determined by the *fft_size* used. We run experiments with *fft_size=256*, which implies that each *fft_block* of 2.0 MHz contains 214 frequency bins. We also set duty_cycle equal to 2 ms. Figure 5(a) shows the correlation between the analog signal acquired by two sensors that are 3 meters away. In the case that the sensors are set to the same frequency band (f_c=101 MHz) the correlation value reaches 0.7 for average factors higher than 10. Note that with a avg_factor of 10, the average is computed every 20 ms. The correlation keeps stable and close to 0 in the case that two sensors are set to a different frequency bands. Therefore, we can reliably detect that the sensors are not reading in the same frequency band. Figure 5(b) shows how we reach a good correlation value between digital signals if the sensors are configured with averaging factor from 20 on-wards. Again the correlation over digital signals when two sensors are set to a different frequencies is stable and close to -0.3.

Impact of Spectrum Resolution. The FFT resolution is an important parameter to set in our system because it will decide, first, the computational cost on the sensor in order to compute the FFT and then the average of the FFT bins, and second the bandwidth used in order to transfer the FFT bins from the sensors to the backend. We would like to reach the maximum correlation value with the minimum data sent to the backend. Figure 6 shows that there is not so



(a) Analog Signal.

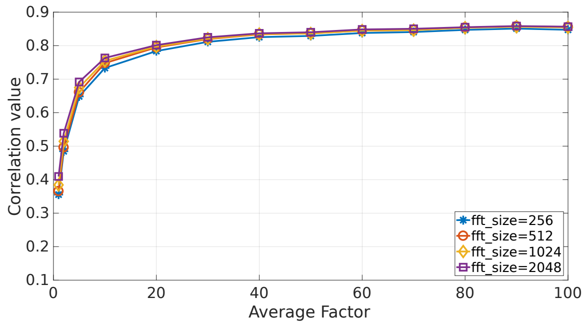


(b) Digital Signal.

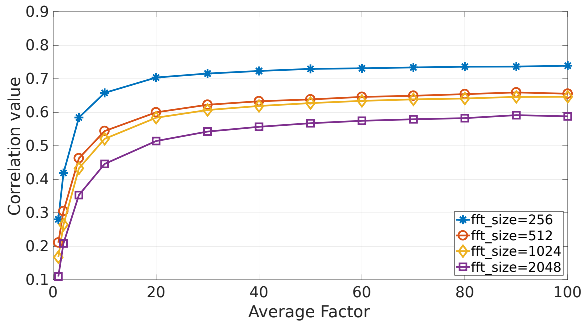
Fig. 5. Analog and digital signal correlation between sensors with different averaging (fft_size=256 and duty_cycle=2ms.)

much difference among using different FFT values (256 up to 2048) in order to reach a good correlation value. However, using digital signals, we can see how the lower is the fft_size the better is the correlation value. For digital signals that change faster that analog ones and contain more information seems to be a good option to use low values of fft_size to reach the maximum correlation between signals. In addition, larger fft_size tend to be affected by larger noise, which can negatively affect the correlation. Both analog and digital signals can be correlated using low values of fft_size (256), which also minimizes the bandwidth used by the sensors to perform the correlation on the backend.

Distance between the Sensors. The distance between the sensors is an important factor that can affect the spectrum correlation among sensors. If the sensors are located far from each other, the channel path loss can severely affect any collaborative strategy. This phenomenon is frequency dependent, since the path loss model decays with $20 \log_{10} f_c$, where f_c indicates the carrier frequency. Therefore higher density is expected at higher frequency to harness the spectrum correlation.



(a) Analog Signal.

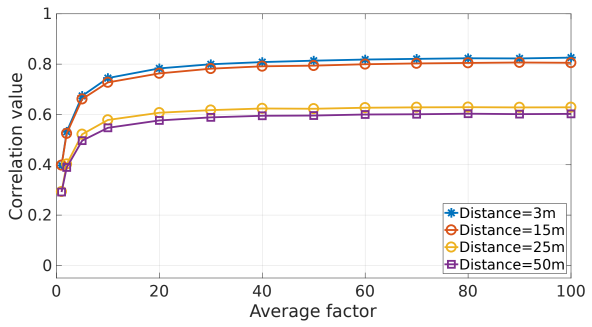


(b) Digital Signal.

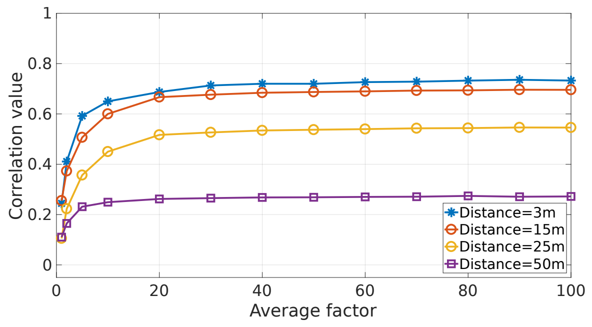
Fig. 6. Analog and digital signal correlation (same frequency band) between sensors with different *fft_size* (duty_cycle=2ms.)

In this experiment we locate several sensors at indoor environment with different distances between each other. The distances between sensors are 3, 15, 25 and 50 meters respectively. We remark that the tests are conducted in indoor environment. Figure 7(a) shows the signal correlation of the sensors located at different distances and scanning an analog signal. Up to 15 meters the correlation value is close to 0.8 and with distances between 25 and 50 meters the correlation is around 0.6. Figure 7(b) shows the same scenario but using digital signals. The correlation for distances up to 15 meters reports a value above 0.6 but for longer distances the correlation decrease dramatically. These results suggest that for analog signals the sensors could work collaboratively together for maximum distances of 50 meters and for digital signals the limit is around 15-20 meters.

Minimum bandwidth required. The goal of this experiment is to provide an answer to the following question: what is the minimum sampling time and the minimum data sent to the backend in order to measure the similarity in the spectrum ? In a distributed approach, like the one studied in this work, it is required to make decisions in the shortest possible time and with the minimum amount of data. Therefore, it is important to evaluate the convergence time to



(a) Analog Signal (101 MHz)



(b) Digital Signal (806 MHz)

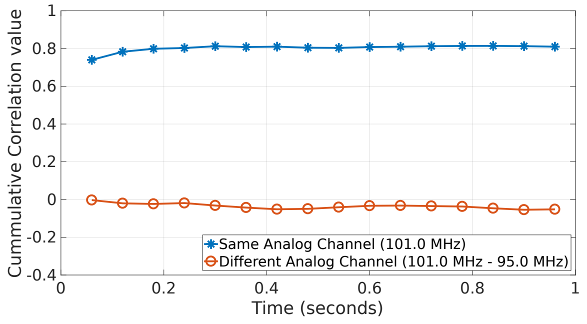
Fig. 7. Signal correlation with different distances between sensors

decide if the sensors are getting similar spectrum or not, and the minimum data sent over Internet to perform the correlation in the backend.

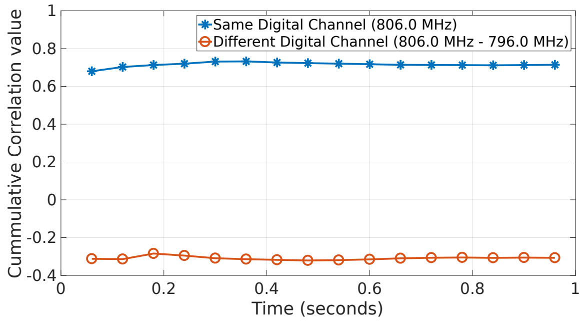
In order to detect when the system converges, we define the *cumulative correlation (CC) metric* as the average of the last K observations of correlation values. Figure 8 shows the CC metric over time for analog and digital signals. We can observe that after 0.2 seconds we reach a stable correlation value that allows us to distinguish if the sensors are reading a similar spectrum or not. Figure 9 shows the data sent to the backend over time for different fft_size configurations. As we determined in the previous experiment, the best configuration of fft_size to perform the correlation in the backend is 256. In addition to that, we have already found that after 0.2 seconds the correlation gets stable. From Fig. 9, we can then conclude that every sensor needs to send only *1 Kbyte* to the backend in order to get a reliable decision about the spectrum similarity.

6 Related Work

Monitoring large spectra is more challenging that conventional approach that sense narrowband frequency spectrum [10–12]. Large corporation have shown



(a) Analog Signal.



(b) Digital Signal.

Fig. 8. Convergence of signal correlation using fft_size=256, avg_factor=30 and duty_cycle=2ms.

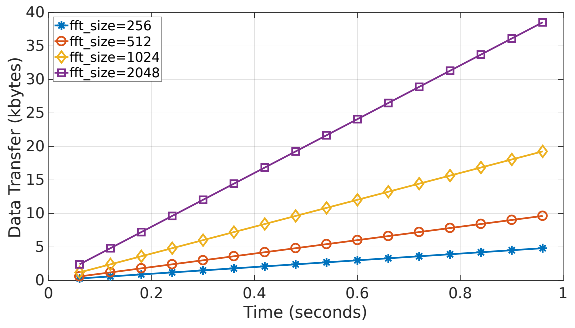


Fig. 9. Data transferred to the server for a duty_cycle=2ms and avg_factor=30.

interest in the topic of large-scale spectrum sensing. Google has lunched the Spectrum Database [13], a joint initiative between Google, industry and regulators to make more spectrum available by using a database to enable dynamic spectrum sharing in TV white spaces. They provide an access to the data to query white space availability based on time and location. Microsoft Spectrum

Observatory [1] is a platform with cost of approximately 5000 dollars that is currently operating in a few locations worldwide. Using data collected with the Spectrum Observatory, [14] proposed a system that identifies transmitters from raw spectrum measurements without prior knowledge of transmitter signatures. It harnesses the observation that wireless signal fading follows a Rayleigh distribution and applies a novel machine learning algorithm and experimentally identifies transmitters robustly. SpecNet [3] is a platform developed by Microsoft Research that allows researches to conduct spectrum measurements studies remotely in real-time for opportunistic spectrum access.
In their recent work, [8] described the implementation and evaluation of a real-time, centralized spectrum monitoring and alert system. These controllers rely on real-time awareness of spectrum activity. However, the analysis is conducted using binary vectors, by comparing each power value in the received power vector to a user-defined threshold. Cooperative sensing has been introduced in [15], and it describes a scenario with sensing nodes that distributively monitor a frequency spectrum. In [16], they proposed to use a cooperative environment to distinguish between an unused band and deep fade due to shadowing or fading effects. The works above did not complement the proposed simulation environment with a system-level design and implementation. [17, 18] employ correlation techniques in different environments, but the study is based on simulations only. No system architecture problems were studied in these work for their actual implementation.
SpecInsight was introduced in [19] and is a system for acquiring 4 GHz of spectrum in real-time using USRP radios with tens of MHz in 7 locations in the US. SpecInsight applies a multi-armed bandit game to the problem of selecting the frequency bands to monitor. Because of the high-end platform used in the work, there are little opportunities for pervasive deployments. In the context of low-cost spectrum sensing systems, [7] and [6] use Raspberry Pi embedded hardware and DVB-T receivers with only 2 MHz of band to monitor 1.7 GHz of spectrum with software defined radio capabilities and implement low-complexity algorithms for distributed spectrum sensing. In addition, [7] proposed different frequency hopping strategies to overcome the hardware limitations of low-cost radios and incorporates different analysis and error/noise correction techniques for flexible and configurable remote wide-spectrum monitoring. In [4], the authors performed an initial feasibility study to verify the efficiency of using RTL-SDR USB dongles connected to laptop or smartphone devices to build a low-cost commoditized spectrum analysis system.

7 Conclusion

We have addressed the compute in the backend the similarity in the spectrum of data collected by low-cost sensors in range of the same transmitter. We have proposed various methods to achieve reliable correlation in presence of noisy data both for analog and digital signals. We have proved that our architecture provides good spectrum correlation for sensors in radio range. We have then studied problems such as how the spectrum correlation varies with the distance between sensors. Our system can reliably correlate the spectrum of analog and digital signals of different sensors using data collected during 200 ms and sending only 1 Kbyte of data per sensor to the backend. Our results are promising and can allow intelligent collaborative decisions among sensors in different locations, even when there is only a coarse information of the physical location of the sensors.

Limitations of the current approach and directions of research

In this work we have shown the feasibility of correlating the spectrum data received by independent sensors. In particular, the study has considered both analog and digital signals, yet with unique transmitter sources (cell tower of FM radio and LTE signal). More in general, medium access protocols and topological issues may require more signal processing and intelligent logic. In particular, multiple transmitters, all of them placed in different locations, may share the same spectrum, yet being in range of the same spectrum sensor. The spectrum sensor could declare high spectrum similarity or not depending on a) the medium access protocol (which transmitter is using the spectrum at a given moment in time) and b) its relative position with respect to the sensors. Possible directions of research to solve this problem are to identify the transmitter, e.g. detecting a specific signature, its signal strength, or even estimating both spectrum similarity and the position of the transmitter. Conclusions could then be drawn about the probability of spectrum similarity based on the spectrum activity of each transmitter.

Acknowledgments

This work has been funded in part by the TIGRE5-CM program (S2013/ICE-2919). We thank the anonymous reviewers for their valuable comments and suggestions.

References

1. “Microsoft spectrum observatory,” http://observatory.microsoftspectrum.com/.
2. J. Naganawa, H. Kim, S. Saruwatari, H. Onaga, and H. Morikawa, “Distributed spectrum sensing utilizing heterogeneous wireless devices and measurement equipment,” in *New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2011 IEEE Symposium on*, May 2011, pp. 173–184.
3. A. Iyer, K. K. Chintalapudi, V. Navda, R. Ramjee, V. Padmanabhan, and C. Murthy, “Specnet: Spectrum sensing sans frontières,” in *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*. USENIX, March 2011.
4. A. Nika, Z. Zhang, X. Zhou, B. Y. Zhao, and H. Zheng, “Towards commoditized real-time spectrum monitoring,” in *Proceedings of the 1st ACM Workshop on Hot Topics in Wireless*, ser. HotWireless ’14. New York, NY, USA: ACM, 2014, pp. 25–30.

5. A. Arcia-Moret, E. Pietrosemoli, and M. Zennaro, “WhispPi: White space monitoring with Raspberry Pi,” *Global Information Infrastructure Symposium, GIIS 2013*, 2013.
6. S. Grnroos, K. Nybom, J. Bjrkqvist, J. Hallio, J. Auranen, and R. Ekman, “Distributed spectrum sensing using low cost hardware,” *Journal of Signal Processing Systems for Signal Image and Video Technology*, p. 113, 2015.
7. D. Pfammatter, D. Giustiniano, and V. Lenders, “A Software-defined Sensor Architecture for Large-scale Wideband Spectrum Monitoring,” in *Proceedings of the 14th International Symposium on Information Processing in Sensor Networks*, ser. IPSN ’15, Seattle, WA, USA, April, pp. 71–82.
8. M. Souryal, M. Ranganathan, J. Mink, , and N. E. Ouni, “Real-time centralized spectrum monitoring: Feasibility, architecture, and latency,” in *Dynamic Spectrum Access Networks (DySPAN), 2015 IEEE Symposium on*, September 2015.
9. R. Calvo-Palomino, D. Giustiniano, , and V. Lenders, “Electrosense: Crowdsourcing spectrum monitoring,” in *2017 IEEE International Conference on Computer Communications (Infocom)*, May 2017.
10. I. F. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, “Next generation/dynamic spectrum access/cognitive radio wireless networks: A survey,” *Comput. Netw.*, vol. 50, no. 13, pp. 2127–2159, Sep. 2006.
11. E. Axell, G. Leus, E. Larsson, and H. Poor, “Spectrum sensing for cognitive radio : State-of-the-art and recent advances,” *Signal Processing Magazine, IEEE*, vol. 29, no. 3, pp. 101–116, May 2012.
12. Z. Q. Z. Quan, S. C. S. Cui, a.H. Sayed, and H. Poor, “Wideband Spectrum Sensing in Cognitive Radio Networks,” *2008 IEEE International Conference on Communications*, 2008.
13. “Google spectrum database,” https://www.google.com/get/spectrumdatabase/.
14. M. Zheleva, R. Chandra, A. Chowdhery, A. Kapoor, and P. Garnett, “Txminer: Identifying transmitters in real-world spectrum measurements,” in *Dynamic Spectrum Access Networks (DySPAN), 2015 IEEE Symposium on*, September 2015.
15. S. Mishra, A. Sahai, and R. Brodersen, “Cooperative Sensing among Cognitive Radios,” *2006 IEEE International Conference on Communications*, pp. 1658–1663, 2006.
16. A. Ghasemi and E. S. Sousa, “Collaborative spectrum sensing for opportunistic access in fading environments,” *2005 1st IEEE International Symposium on New Frontiers in Dynamic Spectrum Access Networks, DySPAN 2005*, pp. 131–136, 2005.
17. A. S. Cacciapuoti, I. F. Akyildiz, and L. Paura, “Correlation-aware user selection for cooperative spectrum sensing in cognitive radio ad hoc networks,” *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 2, pp. 297–306, 2012.
18. M. Sampietro, G. Accomoando, L. G. Fasoli, G. Ferrari, and E. C. Gatti, “High Sensitivity Noise Measurement with a Correlation Spectrum Analyzer,” *IEEE Transactions on Instrumentation and Measurement*, vol. 49, no. 4, pp. 1–3, 2000.
19. L. Shi, P. Bahl, and D. Katabi, “Beyond sensing: Multi-ghz realtime spectrum analytics,” in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, May 2015, pp. 159–172.